MASTER EN SCIENCES INFORMATIQUES

Interface graphique pour un courrier électronique

Loos, Yves; Piron, Philippe

Award date:
1986

Awarding institution:
Universite de Namur

INTERFACE GRAPHIQUE POUR

UN COURRIER ELECTRONIQUE

DIRECTEUR : ROLAND LESUISSE

MEMOIRE PRESENTE PAR

YVES LOOS ET PHILIPPE PIRON

EN VUE DE L'OBTENTION DU

GRADE DE LICENCIE ET MAITRE

EN INFORMATIQUE

ANNEE ACADEMIQUE 1985 - 1986

Nous tenons à remercier monsieur Roland Lesuisse pour l'aide précieuse qu'il nous a apportée durant l'élaboration de ce travail.

Nous tenons à remercier également monsieur Bernard Goebel qui a mis un SUN à notre disposition pour nous permettre d'achever la programmation de ce mémoire.

Enfin, nous remercions madame Therese Collet-Petitjean et monsieur Philippe Delhaye pour leur disponibilité et leurs conseils.

TABLE DES MATIERES

0 INTRODUCTION 1

1 SYNTHESE DU COURRIER ELECTRONIQUE IMPLANTE ... 3

1.1 INTRODUCTION 3

1.2 CONCEPTS 3

1.2.1 COURRIER ELECTRONIQUE 3

1.2.2 ABONNE 3

1.2.3 ADMINISTRATEUR DU C.E. 4

1.2.4 DOCUMENT CIRCULANT 4

1.2.5 SIGNATAIRE D'UN SERVICE 7

1.2.6 DOCUMENTS EN ATTENTE D'UN ACCUSE ..... DE RECEPTION 7

1.2.7 DOCUMENTS EN ATTENTE D'UNE REPONSE ... POUR UN ABONNE 7

1.2.8 BOITE AUX LETTRES D'UN ABONNE 8

1.2.9 TRACE D'UN DOCUMENT 8

1.2.10 COURRIER ARCHIVE POUR UN ABONNE 8

1.2.11 DOCUMENTS EN ATTENTE DE TRAITEMENT ... POUR UN ABONNE 9

1.2.12 DOCUMENTS EN POUBELLE POUR UN ABONNE . 9

1.3 FONCTIONS OFFERTES 9

1.3.1 FONCTIONS OFFERTES AUX ABONNES 9

1.3.1.1 EXPEDITION DE DOCUMENTS 9

1.3.1.2 CONSULTATION ET TRAITEMENT DE ..... DOCUMENTS 9

1.3.1.3 GESTION DU SIGNATAIRE 11

1.3.1.4 FONCTIONS DIVERSES 11

1.3.2 FONCTIONS OFFERTES AUX ADMINISTRATEURS 12

1.4 CONCLUSION 13

2 CONCEPTION D'UN INTERFACE AMELIORE 15

2.1 INTRODUCTION 15

2.2 DEFAUT DANS LA CONCEPTION DE L'IUO 16

2.3 PRE-REQUIS DE LA CONCEPTION D'UN IUO 17

2.3.1 CONNAITRE LA TACHE 17

2.3.2 CONNAITRE LES UTILISATEURS 17

2.4 QUALITES D'UN BON INTERFACE 19

2.4.1 LA COMPATIBILITE 19

2.4.2 L'HOMOGENEITE 20

2.4.3 LA CONCISION 20

2.4.4 LA FLEXIBILITE 20

2.4.5 LE GUIDAGE 21

2.4.6 LA CHARGE INFORMATIONNELLE 21

2.4.7 LE CONTROLE EXPLICITE 21

2.4.8 LA GESTION DES ERREURS 22

2.4.9 CONCLUSION 22

2.5 LE DIALOGUE 22

2.5.1 LE DIALOGUE PROPREMENT DIT 23

2.5.2 LES ENTREES 25

2.5.3 LES SORTIES 28

2.6 INTERFACE D'UN COURRIER ELECTRONIQUE 30

2.7 CONCLUSION 32

3 ENVIRONNEMENT DE BASE 34

3.1 INTRODUCTION 34

3.2 ENVIRONNEMENT HARDWARE 34

3.3 ENVIRONNEMENT SOFTWARE 36

3.3.1 UTILISATION DES FENETRES 36

3.3.1.1 LES FENETRES 36

3.3.1.2 LES SOUS-FENETRES 40

3.3.1.3 LES ICONES 43

3.3.1.4 LES MENUS 45

3.3.1.5 LES PANNEAUX DE COMMANDE 52

3.3.1.6 DEMARRAGE D'UN TOOL 53

3.3.2 PROGRAMMATION DES FENETRES 53

3.3.2.1 OUTILS DISPONIBLES 53

3.3.2.2 TECHNIQUES DE PROGRAMMATION DES .... TOOLS 56

3.4 CONCLUSION 61

4 INTERFACE DEVELOPPE 63

4.1 INTRODUCTION 63

4.2 PROBLEMES DE L'INTERFACE DE DEPART 63

4.3 CARACTERISTIQUES DE L'INTERFACE AMELIORE . 65

4.3.1 ECRAN STATIQUE 65

4.3.2 EDITEUR INTERNE AU PROGRAMME 65

4.3.3 SIMPLICITE DES COMMANDES 66

4.3.4 TACHES EXECUTEES EN PARALLELE 67

4.4 STRUCTURE GENERALE DE L'ECRAN 67

4.5 DESCRIPTION DES SOUS-FENETRES 69

4.5.1 SOUS-FENETRE DES MESSAGES 69

4.5.2 SOUS-FENETRE DES PARAMETRES 72

4.5.3 SOUS-FENETRE DES DOCUMENTS 75

4.5.4 SOUS-FENETRE DES RESUMES 77

4.5.5 SOUS-FENETRE DE COMMANDES 79

4.5.5.1 LES COMMANDES PORTANT SUR UN ICONE . 81

4.5.5.2 LES COMMANDES NE PORTANT PAS SUR ... UN ICONE 81

4.5.6 SOUS-FENETRE DES ICONES 84

4.5.6.1 LES ICONES REPRESENTANT UNE ..... COLLECTION DE DOCUMENTS 84

4.5.6.2 LES AUTRES ICONES 84

4.5.6.3 REGLE GENERALE 87

4.6 RESPECT DES QUALITES D'UN BON INTERFACE .. 88

4.6.1 RESPECT DES PRE-REQUIS 88

4.6.2 RESPECT DES QUALITES 89

4.6.2.1 COMPATIBILITE 89

4.6.2.2 HOMOGENEITE 89

4.6.2.3 CONCISION 90

4.6.2.4 FLEXIBILITE 90

4.6.2.5 GUIDAGE 90

4.6.2.6 CHARGE INFORMATIONNELLE 91

4.6.2.7 CONTROLE EXPLICITE 91

4.6.2.8 GESTION DES ERREURS 91

4.6.3 RESPECT DES RECOMMANDATIONS POUR LE DIALOGUE 92

4.6.4 RESPECT DES RECOMMANDATIONS POUR UN COURRIER ELECTRONIQUE 93

4.7 EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE 93

4.7.1 INTRODUCTION 93

4.7.2 DEROULEMENT DE L'EXECUTION 94

4.8 CONCLUSION 95

5 ANALYSE ORGANIQUE 97

5.1 INTRODUCTION 97

5.2 PRINCIPES GENERAUX DE L'IMPLEMENTATION ... 98

5.3 DICTIONNAIRE DES VARIABLES 104

5.4 SPECIFICATION DES FONCTIONS PRINCIPALES .. 108

5.4.1 IMPMES 108

5.4.2 SIGC 109

5.4.3 INIT_FEN 109

5.4.4 FEN_SIGHANDLER 110

5.4.5 IMPRIM_LISTE 111

5.4.6 GES_MENU_CONS_RES 112

5.4.7 IMPRIM_CHAR_SUIV_FEN 113

5.4.8 FEN_INPUT 114

5.4.9 TRAIT_ARCHIVE 115

5.4.10 INIT_OPTION1 117

5.4.11 C301_PROC 118

5.4.12 C202_PROC 119

6 MODE D'EMPLOI 120

6.1 INTRODUCTION 120

6.2 LANCEMENT DU PROGRAMME 121

6.3 ACCES AU COURRIER ELECTRONIQUE 121

6.4 PRINCIPES GENERAUX D'UTILISATION 121

6.5 POSSIBILITE DE TRAVAILLER EN PARALLELE ... 123

6.6 DEFINITION DES COMMANDE-MENUS 124

6.7 LES CONSULTATIONS 134

6.7.1 CONSULTATION DE LA POUBELLE 134

6.7.2 CONSULTATION DES ARCHIVES 135

6.7.3 CONSULTATION DES DOCUMENTS EN ATTENTE 138

6.7.3.1 CONSULTATION DES DOCUMENTS EN ..... ATTENTE DE TRAITEMENT 138

6.7.3.2 CONSULTATION DES DOCUMENTS EN ..... ATTENTE DE REPONSE 138

6.7.3.3 CONSULTATION DES DOCUMENTS EN ..... ATTENTE D'UN ACCUSE DE RECEPTION ... 139

6.7.4 CONSULTATION DE LA BOITE-IN 139

6.7.5 CONSULTATION DU SIGNATAIRE 139

6.7.6 CONSULTATION DE LA POSTE 140

6.7.7 CONSULTATION MOT DE PASSE 140

6.7.8 CONSULTATION DES REPONSES RECUES 141

6.7.9 CONSULTATION DU DOCUMENT PERE 142

6.7.10 CONSULTATION DE L'ADMINISTRATION 142

6.7.11 CONSULTATION DU BOTTIN 143

6.8 TRANSFERT 143

6.8.1 TRANSFERT VERS LA BOITE-OUT 143

6.8.2 TRANSFERT VERS L'IMPRIMANTE 145

6.8.3 LES AUTRES TRANSFERTS 145

6.9 EDITION 145

6.10 SORTIE DU PROGRAMME 146

7 CONCLUSION 147

BIBLIOGRAPHIE 149

TABLE DES FIGURES

Figure 2.1: REPONSE APPARAISSANT A COTE DU DOCUMENT LA MOTIVANT 31

Figure 3.1: SOURIS DU SUN 36

Figure 3.2: TOOLS STANDARDS SHELLTOOL ET CLOCKTOOL 39

Figure 3.3: DEPLACEMENT DES FENETRES 41

Figure 3.4: SOUS-FENETRES 44

Figure 3.5: ICONES 46

Figure 3.6: MENUS 48

Figure 3.7: MENU ROOT-MANAGER 50

Figure 3.8: MENU TOOL-MANAGER 51

Figure 3.9: PANNEAU DE COMMANDE 52

Figure 3.10: NIVEAUX DES LIBRAIRIES DU SYSTEME DES FENETRES ... 54

Figure 3.11: STRUCTURE GENERALE D'UN TOOL 57

Figure 4.1: STRUCTURE GENERALE DE L'ECRAN 68

Figure 4.2: SOUS-FENETRES DES MESSAGES 70

Figure 4.3: EXEMPLE D'UTILISATION DE LA SOUS-FENETRE DES MESSAGES 71

Figure 4.4: EXEMPLE D'UTILISATION DE LA SOUS-FENETRE DES PARAMETRES 72

Figure 4.5: SOUS-FENETRE DES PARAMETRES 73

Figure 4.6: SOUS-FENETRE DES DOCUMENTS 76

Figure 4.7: SOUS-FENETRE DES RESUMES 78

Figure 4.8: SOUS-FENETRE DES COMMANDES 80

Figure 4.9: SOUS-FENETRE D'EDITION 83

Figure 4.10: SOUS-FENETRE DES ICONES 85

Figure 4.11: ICONES REPRESENTANT UNE COLLECTION DE DOCUMENTS .. 86

Figure 4.12: ICONES NE REPRESENTANT PAS UNE COLLECTION DE DOCUMENTS 87

Figure 4.13: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN ... MESSAGE 96/2

Figure 4.14: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN ... MESSAGE 96/3

Figure 4.15: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN ... MESSAGE 96/4

Figure 4.16: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN ... MESSAGE 96/5

Figure 4.17: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN ... MESSAGE 96/6

0 INTRODUCTION

Un des aspects les plus importants à l'heure actuelle dans la conception de logiciels est l'interface entre l'utilisateur et le programme. En effet, le nombre de domaines dans lesquels l'informatique s'introduit augmente de jour en jour, et ses nouveaux utilisateurs sont de moins en moins des spécialistes en ce qui concerne le fonctionnement non seulement des ordinateurs, mais également des programmes qu'ils exécutent. C'est pour cette raison qu'un bon interface est important, car il permet à des personnes qui ne sont pas ou peu initiées d'utiliser un logiciel d'une manière plus efficace.

L'objectif de ce mémoire va dans ce sens. En effet, il consiste en l'amélioration de l'interface d'un logiciel de courrier électronique qui a été réalisé l'année dernière dans le cadre d'un mémoire. Cette amélioration a pour but principal de rendre ce logiciel plus facile à utiliser.

Le matériel utilisé pour réaliser le premier mémoire, un réseau d'I.B.M. P.C., ne permettait pas de faire un interface très puissant. Par contre, le matériel mis à notre disposition pour le développement de ce mémoire est un matériel de pointe en ce qui concerne l'interfaçage. En effet, le Sun est un micro-ordinateur de fabrication récente, qui dispose notamment d'un écran de grande taille, d'une grande capacité mémoire et d'outils logiciels puissants.

Nous commencerons par rappeler en quoi consiste le mémoire de courrier électronique qui est à la base de notre mémoire car nous avons conservé l'ensemble de ses fonctions [chapitre 1]. Ensuite, nous présenterons les principes généraux de conception d'un interface amélioré [chapitre 2]. Après cela, nous décrirons les outils dont nous nous sommes servis pour réaliser ce mémoire, tant au point de vue du matériel qu'au point de vue des logiciels [chapitre 3]. Cette description est nécessaire vu le fait que ces outils sont différents de ceux que l'on utilise habituellement, et notamment de ceux qui furent utilisés pour réaliser le premier mémoire. Après ces trois premiers chapitres qui décrivent sur quoi nous nous sommes basés pour réaliser notre interface, nous présenterons la solution que nous avons choisie pour améliorer l'interface de départ [chapitre 4]. Il existait en effet plusieurs solutions pour résoudre les problèmes posés, et nous avons opté pour celle qui facilitait le plus possible la tâche de l'utilisateur, au détriment parfois de la puissance d'utilisation. Pour terminer, nous avons décrit les principes généraux de programmation que les outils impliquaient et nous avons spécifié les fonctions de notre programme [chapitre 5], tout ceci étant suivi du mode d'emploi [chapitre 6], qui est très différent de celui du premier mémoire.

1 SYNTHESE DU COURRIER ELECTRONIQUE IMPLANTE

1.1 INTRODUCTION

Ce chapitre est une synthèse de l'analyse fonctionnelle du "Courrier électronique" [A. JOSIS, P. BERNARD] qui nous a servi de point de départ et dont nous avons modifié l'interface. Nous commencerons d'abord par définir les principaux concepts qui interviennent dans ce courrier électronique et auxquels nous nous référerons ultérieurement. Puis nous décrirons les fonctions qu'il offre.

1.2 CONCEPTS

1.2.1 COURRIER ELECTRONIQUE

On appelle courrier électronique (C.E.) un système automatique de transmission et de distribution à des abonnés de documents circulant à l'intérieur d'une organisation.

1.2.2 ABONNE

On appelle abonné tout individu ou service auquel il est permis de transmettre, recevoir et faire distribuer des documents

par le C.E..

Tout abonné est caractérisé obligatoirement par :

- un identifiant d'abonné parmi tous les abonnés et un mot de passe qui lui permettent de travailler avec le C.E. ;
- sa fonction à l'intérieur de l'organisation (service, patron, secrétaire ou dactylo) ;
- le service auquel il appartient (facultatif cependant pour un patron) ;
- un mot de passe de confidentialité qui lui permet de protéger les documents confidentiels qu'il reçoit ou expédie.

1.2.3 ADMINISTRATEUR DU C.E.

On appelle administrateur du C.E. tout abonné ayant accès aux fonctions d'administration du C.E.

1.2.4 DOCUMENT CIRCULANT

On appelle document circulant tout texte transmis d'un abonné (expéditeur) à un ou plusieurs autres abonnés (destinataires); et par texte, on entend notes de service, courrier général, rapports, dossiers à l'exclusion de photos, de graphiques ou de messages vocaux ou contenant des caractères spéciaux, ... L'expéditeur d'un document peut en être destinataire à condition qu'il soit le seul destinataire. Le C.E. peut, de cette manière, être utilisé en tant qu'agenda.

Tout document expédié a quelques attributs qu'on va définir :

- type d'un document : le type d'un document est une combinaison, même vide, des caractéristiques de document suivantes :
  - recommandé : un document recommandé est un document pour lequel l'expéditeur exige un accusé de réception de tous les destinataires. L'accusé de réception d'un document par un destinataire marque le fait que ce destinataire est au courant de la réception de ce document ;
  - demande de réponse : une demande de réponse est un document pour lequel l'expéditeur exige une réponse de la part de tous les destinataires ;
  - réponse : une réponse est un document qui répond à un autre ;
  - rappel : un rappel est un document par lequel l'expéditeur réclame de tous les destinataires qui n'ont pas encore répondu, une réponse à une demande de réponse précédemment envoyée ;
  - confidentiel : un document confidentiel est un document dont la lecture ou la relecture, tant par l'un des destinataires que par l'expéditeur, exige la fourniture du mot de passe de confidentialité associé à ce destinataire ou à l'expéditeur.
- structure d'un document : chaque document est constitué :
  - d'une entête où l'on trouve les informations suivantes :
    - l'identifiant du document reprenant l'identifiant d'abonné de l'expéditeur, la date d'expédition et

un numéro de séquence (...) ;

- le type du document ;
- le titre du document qui est le plus souvent un résumé du contenu ;
- l'identifiant d'abonné du rédacteur du document qui peut être différent de celui de l'expéditeur ;
- si le document est une réponse, l'identifiant de la demande de réponse à laquelle ce document répond ;
- si le document est une demande de réponse, la date ultime pour y répondre (dans la suite, on appellera cette date "la date d'échéance") ;
- la liste des destinataires : cette liste est prédéfinie (utilisation d'une liste prédéfinie de destinataires) ou propre à un envoi ;
- une note éventuelle qui indique aux destinataires l'action à effectuer sur le document ;
- une liste éventuelle de mots-clés qui permet à l'expéditeur et aux destinataires de retrouver un document archivé.

- d'un corps ou texte proprement dit ;
- d'une clôture où l'on trouve les informations suivantes :
  - la liste des accusés de réception déjà reçus si le document expédié était recommandé, avec pour chaque accusé de réception, l'identifiant du destinataire concerné et la date d'envoi de cet accusé de réception (...) ;
  - la liste des réponses déjà reçues si le document

expédié était une demande de réponse, avec pour chaque réponse, l'identifiant du destinataire, l'identifiant du document réponse et la date du dernier rappel.

1.2.5 SIGNATAIRE D'UN SERVICE

Le signataire d'un service est la collection des documents soumis à l'approbation et à la signature du patron de ce service par l'ensemble des abonnés de ce service.

1.2.6 DOCUMENTS EN ATTENTE D'UN ACCUSE DE RECEPTION

C'est la collection des documents recommandés expédiés par cet abonné et pour lesquels les accusés de réception n'ont pas encore été reçus de tous les destinataires.

1.2.7 DOCUMENTS EN ATTENTE D'UNE REPONSE POUR UN ABONNE

C'est la collection des demandes de réponse expédiées par cet abonné et pour lesquelles des réponses n'ont pas encore été reçues de tous les destinataires.

1.2.8 BOITE AUX LETTRES D'UN ABONNE

C'est la collection des documents reçus par cet abonné et qu'il n'a pas encore retirés de sa boite. Il peut déjà être au courant de la réception d'un document en boite et même en avoir pris connaissance (on parle alors d'un ancien document en boite). Par contre, il peut encore ignorer la réception d'un document en boite (on parle alors d'un nouveau document en boite).

1.2.9 TRACE D'UN DOCUMENT

On appelle trace d'un document expédié via le C.E., une information constituée de :

- l'identifiant du document ;
- le type du document ;
- la liste des destinataires, avec pour chacun d'eux la date à laquelle il a reçu le document.

1.2.10 COURRIER ARCHIVE POUR UN ABONNE

C'est la collection des documents expédiés par cet abonné qui n'attendent pas (plus) un accusé de réception ou une réponse et qui ont donc pu être archivés par lui, augmentée des documents reçus par cet abonné et qu'il a décidé d'archiver.

1.2.11 DOCUMENTS EN ATTENTE DE TRAITEMENT POUR UN ABONNE

C'est la collection des documents reçus par cet abonné, qu'il a lus et dont il a différé les traitements à des dates au plus tard.

1.2.12 DOCUMENTS EN POUBELLE POUR UN ABONNE

C'est la collection des documents reçus par cet abonné, qu'il a lus et dont il ne désire pas garder de trace.

1.3 FONCTIONS OFFERTES

1.3.1 FONCTIONS OFFERTES AUX ABONNES

1.3.1.1 EXPEDITION DE DOCUMENTS

Un utilisateur peut expédier un document qu'il vient d'éditer, ou qui est en signataire, ou qui a déjà été envoyé.

1.3.1.2 CONSULTATION ET TRAITEMENT DE DOCUMENTS

Il existe huit collections de documents :

- la boite aux lettres ;

- les documents en attente d'un accusé de réception ;
- les documents en attente de réponse ;
- les documents en attente de traitement ;
- les réponses reçues à un document envoyé ;
- le document reçu qui est à l'origine d'un document envoyé ;
- le courrier archivé ;
- la poubelle.

Dans chaque collection, il est possible de consulter soit les documents, soit les résumés de ces documents, et de sélectionner le document sur lequel on va travailler (il existe une exception à cette méthode : pour la consultation du courrier archivé, on utilise une combinaison de valeurs de critères tels que l'identifiant du document, la date d'envoi de ce document, une liste de mots-clés ...).

Une fois un document sélectionné, il est possible de faire une série d'opérations le concernant comme par exemple l'imprimer, le transférer dans une autre collection (quand c'est possible), l'expédier après l'avoir éventuellement modifié et consulter d'autres documents en rapport avec le premier (par exemple pour un document en attente d'un accusé de réception, si ce document est une réponse, on peut consulter le document reçu qui a demandé l'envoi de cette réponse). Le programme gèrera alors les transferts éventuels d'une collection à l'autre et les modifications éventuelles des caractéristiques de ce document.

1.3.1.3 GESTION DU SIGNATAIRE

Le signataire est une collection de documents un peu spéciale. En effet, pour pouvoir y accéder, il faut que l'utilisateur soit le patron du service. De plus, les opérations que l'on peut effectuer sur le document sélectionné (la sélection se fait de la même manière que pour les autres collections) sont différentes. On peut en effet :

- supprimer le document ;
- modifier le document ;
- signer le document ;
- imprimer le document.

Si le traitement effectué est une suppression, le document disparait de la collection des documents en signataire.

Pour pouvoir signer, l'utilisateur doit être le patron du service.

1.3.1.4 FONCTIONS DIVERSES

Ces fonctions sont :

- obtenir des services de la poste, c-à-d consulter la liste des abonnés, ou obtenir la date de réception d'un document qu'il a envoyé ;
- changer ses mots de passe (de confidentialité ou de

connection) ;

- gérer ses listes prédéfinies de destinataires, c-à-d créer, modifier, supprimer, consulter ou imprimer une liste, ou encore consulter la liste des listes de destinataires.

1.3.2 FONCTIONS OFFERTES AUX ADMINISTRATEURS

Ces fonctions s'appellent les fonctions d'administration. Elles sont au nombre de cinq :

- une fonction de gestion journalière du C.E. qui doit être activée au départ de chaque nouveau jour ouvrable mais une seule fois par jour ouvrable. Cette fonction permet de passer d'un jour ouvrable à un autre, de consulter les anomalies (mots de passe erronnés, ...), de supprimer tous les documents qui peuvent l'être (documents depuis 8 jours en poubelle, ...), et de constituer des statistiques pour le jour que l'on quitte (nombre moyen de caractères, nombre moyen de destinataires, nombre de documents envoyés) ;
- une fonction d'exploitation des statistiques constituées. Cette fonction permet, sur une période donnée, d'obtenir soit toutes les dates au cours desquelles un abonné donné a expédié des documents et le nombre associé à chacune de ces dates, soit, pour chaque jour ouvrable de cette période, le nombre moyen de caractères et le nombre moyen de destinataires des documents expédiés ce jour-là ;
- une fonction qui permet de changer le mot de passe d'accès à ces fonctions d'administration ;

- une fonction de gestion des abonnés au C.E.. Cette fonction permet d'ajouter un abonné, d'en supprimer un, et d'obtenir ou de modifier la description d'un abonné ;
- une fonction qui permet de changer la valeur de la clé de codage qui sert à encrypter les documents confidentiels expédiés via le C.E.

1.4 CONCLUSION

L'ensemble de ces fonctions a été implémentée en PASCAL sur un réseau d'I.B.M./P.C.. L'interface utilisé dans ce mémoire est constitué d'une série de menus, une commande d'un menu ayant pour effet d'afficher un menu plus précis jusqu'à ce qu'on arrive à la commande même que l'on veut faire exécuter. Prenons par exemple un utilisateur qui désire répondre à un document qui lui a été envoyé. Après avoir sélectionné la boite aux lettres dans le premier menu, l'utilisateur a le choix dans un deuxième menu entre consulter les résumés et feuilleter les documents. Son choix fait, supposons consulter les résumés, l'utilisateur sélectionne le résumé qui correspond au document auquel il désire répondre. Ensuite, un troisième menu apparait, qui reprend les différents traitements que l'utilisateur peut effectuer sur ce document :

1 : Retour au menu supérieur

2 : Le mettre en attente de traitement

3 : L'archiver

4 : Le faire circuler

5 : L'imprimer

6 : Le mettre à la poubelle

7 : Y répondre

8 : Consulter le document père

L'utilisateur répondra à ce menu par le chiffre 7, pour répondre au document qu'il a sélectionné. A ce moment, si ce document demande une réponse (si ce n'est pas le cas, un message d'erreur avertira l'utilisateur), un quatrième menu apparaitra, demandant si la réponse sera éditée via l'éditeur interne, ou via un autre éditeur.

Notre mémoire a pour objectif de modifier cet interface, en gardant les mêmes fonctions. Ces modifications seront effectuées sur une machine, le SUN, qui permet, grâce à ses qualités techniques et à des logiciels évolués, de réaliser des interfaces plus performants.

2 CONCEPTION D'UN INTERFACE AMELIORE

2.1 INTRODUCTION

Le but de ce mémoire étant d'améliorer l'interface d'un courrier électronique, il nous est paru nécessaire de pouvoir, non pas définir exactement ce qu'est LE BON interface, mais de donner certaines qualités d'un bon interface ["Guide ergonomique de conception des interfaces hommes-ordinateurs" (IUO : interface utilisateur-ordinateur) de Dominique L. Scapin], certains principes et recommandations à suivre. Nous parlerons d'abord des défauts dans la conception d'interfaces. Ensuite, nous poursuivrons par les pré-requis de la conception d'un interface, suivis des qualités d'un bon interface. Nous parlerons aussi du dialogue utilisateur-ordinateur en général, et puis plus particulièrement des entrées et des sorties. Enfin, nous terminerons par l'énumération des règles qui nous paraissent devoir être respectées dans la conception d'un interface pour une application de courrier électronique.

2.2 DEFAUT DANS LA CONCEPTION DE L'IUO

Il existe plusieurs défauts chez les concepteurs d'IUO. En voici les principaux :

- il y a généralement un manque de connaissance des tâches. Certaines séquences de commandes du logiciel ne correspondent pas aux séquences des actions à faire pour exécuter une même tâche ;
- il y a un manque de connaissance des utilisateurs. Le niveau d'expérimentation de ceux-ci est très important pour déterminer le niveau de simplicité de l'interface développé ;
- les tests du logiciel se font rarement sur des scénarios réalistes de tâches ;
- on trouve un manque d'homogénéité dans la présentation des informations (incohérence de dénomination, de structure et d'ordre de séquence) ;
- on constate que les concepteurs ne prévoient pas toujours les erreurs humaines possibles (soit erreurs d'exécution de tâches , soit erreurs de manipulation) ;
- on vise les performances plutôt que la facilité pour l'utilisateur d'exécuter les tâches.

Tous ces défauts sont donc à éviter lors de la création d'un interface, car ils conduisent à une mauvaise utilisation du système. Suite à cela, les performances diminuent, les utilisateurs se plaignent et on en arrive à, soit une utilisation minimale du système avec un retour aux tâches mécaniques lorsque

c'est possible, soit à une déformation de la tâche.

2.3 PRE-REQUIS DE LA CONCEPTION D'UN IUO

Avant de concevoir des interfaces, il est absolument nécessaire de connaître la tâche pour laquelle cet interface sera réalisé, et les utilisateurs potentiels du logiciel.

2.3.1 CONNAITRE LA TACHE

La connaissance de la tâche est très importante pour au moins une raison évidente : si le logiciel veut effectuer CETTE tâche - et non une tâche qui lui ressemble -, il est primordial que celle-ci soit définie avec précision. Dans le cas contraire, une fois le logiciel conçu, la tâche automatisée risque d'être différente de la tâche de départ. Et même si elles ne sont pas différentes, les séquences d'actions pour y parvenir ne seront pas équivalentes, et l'utilisateur sera dérouté du fait qu'il est habitué à agir autrement.

2.3.2 CONNAITRE LES UTILISATEURS

L'interface développé sera fonction des personnes qui s'en serviront. Il existe plusieurs types d'utilisateurs, il doit donc exister plusieurs types d'interfaces. Parmi les utilisateurs potentiels d'un interface, nous pouvons recenser les trois types

suivants :

- ceux qui connaissent l'informatique ;
- ceux qui sont experts dans leur tâche ;
- les novices et les naïfs (c-à-d les non-expérimentés), qui forment la part la plus importante des utilisateurs.

Il va de soi qu'un interface pour un utilisateur expérimenté ne sera pas le même que pour celui qui débute en la matière. Pour ce dernier, le niveau de facilité sera beaucoup plus élevé pour lui permettre de faire le moins d'erreur possible, lui qui sera plus enclin à en faire.

Les utilisateurs les plus nombreux étant ceux qui sont les moins expérimentés, il ne nous paraît pas superflu de donner quelques recommandations pour la conception de logiciels destinés à cette classe d'utilisateur :

- l'initiative de tout dialogue doit venir du logiciel ;
- chaque entrée (commande ou information) doit être brève ;
- les procédures d'entrée d'informations ou de commandes ne doivent pas requérir de formation particulière ;
- pour les entrées à risque (exemple : commandes de suppression ou de destruction) , le logiciel doit demander une confirmation de la part de l'utilisateur ;
- les messages destinés à l'utilisateur doivent être clairs et sans équivoque (pas d'information superflue) ;
- l'utilisateur doit être capable de contrôler le rythme du dialogue (exemple : l'ordinateur ne demande une nouvelle

information à l'utilisateur que lorsqu'il a répondu à la précédente).

2.4 QUALITES D'UN BON INTERFACE

En règle générale, on peut dire que le plus important est de respecter les utilisateurs au niveau de leurs connaissances, de leurs objectifs et de leurs méthodes. Nous allons donc donner certaines qualités que doit avoir un interface s'il veut satisfaire ces utilisateurs (cette liste est non exhaustive).

2.4.1 LA COMPATIBILITE

Dans l'interface, les écrans doivent être compatibles avec les supports papiers et avec la réalité que connaissait jusqu'alors l'utilisateur. De préférence, on emploiera le vocabulaire qui lui est familier. Un changement dans ses habitudes pourrait nuire aux performances et même au bon fonctionnement du logiciel. Profiter au maximum des connaissances acquises ne peut être que bénéfique pour le bon déroulement des opérations.

2.4.2 L'HOMOGENEITE

Il faut que les mêmes actions à exécuter par l'utilisateur suivent toujours le même chemin, c'est-à-dire la même séquence de commandes. Devoir exécuter une tâche de deux manières différentes selon les cas ne peut qu'embrouiller celui qui l'exécute et augmenter le nombre d'erreurs possibles. L'homogénéité diminue donc considérablement les possibilités d'erreurs en familiarisant l'utilisateur avec la façon d'exécuter sa tâche.

2.4.3 LA CONCISION

Il faut faire en sorte que l'utilisateur doive se servir le moins possible de sa mémoire. S'il ne doit pas retenir de longues commandes ou de longues séquences de commandes, les performances et le taux d'erreur n'en seront qu'améliors.

2.4.4 LA FLEXIBILITE

L'interface sera meilleur s'il peut s'adapter aux types d'utilisateurs qui s'en servent. Pour certains logiciels, il n'existera qu'une seule classe d'utilisateurs, tandis que pour d'autres, il en existera plusieurs (plus ou moins expérimenté). De plus, les novices, à force de s'en servir, vont devenir de plus en plus expérimentés, et demanderont un interface plus performant. Il faut donc prévoir une adaptation de l'interface au type d'utilisateurs qui travaillent.

2.4.5 LE GUIDAGE

Lorsqu'une commande a été entrée, il est tout-à-fait essentiel que le système donne une confirmation de l'exécution de la commande ou un message pour interdire d'exécuter celle-ci. Il est aussi nécessaire d'informer soit du succès, soit de l'échec de cette commande. L'utilisateur doit toujours savoir où il se trouve dans la réalisation de ses tâches.

2.4.6 LA CHARGE INFORMATIONNELLE

Il faut essayer qu'il y ait le moins d'opérations possibles à effectuer et le moins de touches à enfoncer. Car plus on minimise le nombre d'opérations à effectuer et donc le nombre d'informations à donner, moins on a de chance de se tromper.

2.4.7 LE CONTROLE EXPLICITE

Il faut toujours donner l'illusion à l'utilisateur que c'est lui qui a le contrôle du dialogue. Chaque commande qu'il donne devra donc provoquer une action par l'ordinateur.

2.4.8 LA GESTION DES ERREURS

On doit minimiser le nombre de possibilités d'erreurs offertes à l'utilisateur. Il faut aussi lui fournir un moyen de détecter ses erreurs et de pouvoir les corriger.

2.4.9 CONCLUSION

Si on regarde l'ensemble de tous ces principes qui devraient mener à la conception d'un bon interface, on remarque que dans la plupart de ceux-ci, le but est d'essayer de minimiser le nombre de possibilités d'erreurs. On s'aperçoit aussi que l'on cherche à ce que l'utilisateur ne soit pas dépaysé par rapport à la façon dont il exécutait son travail auparavant. On essaye donc de maintenir une certaine harmonie entre le travail que l'utilisateur effectuait et le travail qu'il effectuera. Tout cela devrait mener, grâce aussi aux possibilités d'erreurs réduites, à une meilleure utilisation du système.

2.5 LE DIALOGUE

L'IUO suppose toujours un dialogue entre l'utilisateur et l'ordinateur. Nous allons d'abord parler du dialogue proprement dit sans faire de distinction entre les messages du système et les informations entrées par l'utilisateur. Ensuite, nous verrons les entrées et les sorties, chacune séparément. Nous énoncerons enfin des recommandations plus précises sur des aspects du dialogue.

2.5.1 LE DIALOGUE PROPREMENT DIT

Pour faciliter le travail de l'utilisateur, le dialogue doit être à l'initiative de l'ordinateur, c'est-à-dire que c'est lui qui dirige les transactions, c'est lui qui propose des choix de commandes à l'utilisateur, qui ne sera dès lors pas obliger de se les rappeler.

Le type de dialogue doit être fonction du type d'utilisateurs. Comme pour l'interface en général, le dialogue pour les non-expérimentés ne sera pas le même que pour ceux qui le sont. Pour ces derniers, il est préférable de leur permettre de sauter des étapes qui leur font perdre du temps, et leur donner la possibilité de se servir de fonctions plus complexes mais nettement plus performantes.

Une recommandation importante est d'éviter que l'utilisateur ne se trompe. Pour cela, il faut premièrement utiliser les mêmes dialogues pour les mêmes actions. Deuxièmement, il faut simplifier au maximum, c'est-à-dire essayer de limiter au maximum le nombre de commandes. Cela veut dire que les commandes seront plus puissantes et que les performances du système seront dès lors meilleures.

Le type de dialogue qui offre le plus de sécurité est le dialogue par menu. Dans celui-ci, l'ordinateur propose à l'utilisateur plusieurs possibilités de commandes parmi lesquelles

ce dernier doit choisir à l'exclusion de toute autre possibilités.

Exemple de menu :

- (a)jouter une lettre
- (s)upprimer une lettre
- (m)ettre une lettre en majuscule
- (r)etour à un menu plus haut dans une hiérarchie de menus

Lorsque l'ordinateur propose ce menu à l'utilisateur, celui-ci n'a le choix que parmi ces quatre commandes.

Ces menus vont servir d'aide-mémoires à l'utilisateur qui ne devra plus retenir le nom de certaines commandes compliquées ou d'une séquence de commandes plus ou moins longue. Ils permettront aussi d'éviter les entrées erronées d'informations. Pour que ce soit facile à utiliser, il faut ne permettre qu'une seule sélection par menu. De plus, le choix de l'utilisateur sera enregistré par l'entrée de la lettre initiale de la commande choisie. Les menus sont une bonne méthode pour les commandes, mais ne sont pas très adéquats pour l'entrée de paramètres car le choix est limité aux seules composantes de ce menu. Enfin, l'ordre des possibilités offertes dans le menu doit être l'ordre d'utilisation de ces possibilités, ou s'il n'y en a pas, l'ordre de fréquence d'utilisation de celles-ci. En effet, si une commande fortement utilisée se trouve à la fin de la liste des choix du menu, l'utilisateur devra d'abord chaque fois lire tous les autres choix qui précèdent le sien dans la liste.

Un autre type de dialogue consiste à utiliser des touches fonctions. Celles-ci diminuent certes les erreurs de frappe qui sont effectivement nombreuses, mais d'un autre coté, une commande correspond à une touche et les erreurs ont des conséquences beaucoup plus importantes. Par exemple, lorsqu'on travaille dans certains éditeurs, des touches ont été définies de telle sorte que la simple pression de celles-ci ait pour effet la suppression d'un mot, d'un paragraphe ou même d'un texte.

Un dernier point important est l'existence des valeurs par défaut. Celles-ci évitent à l'utilisateur de devoir entrer souvent les mêmes valeurs de paramètres lorsque celles-ci varient très peu.

2.5.2 LES ENTREES

La première recommandation importante à faire aux concepteurs de logiciels d'interfaces est d'essayer de limiter au maximum le nombre de données que l'utilisateur doit entrer. Il y a plusieurs raisons à cela : premièrement, moins il entrera de données, moins il fera d'erreurs; deuxièmement, moins il aura de données à taper, plus l'interface sera facile pour lui. Pour cela, plusieurs principes sont de mise :

- il ne faut pas lui demander d'entrer plusieurs fois les mêmes données car taper plusieurs fois les mêmes choses peut finir par l'agacer et diminuer la qualité et le rendement de son

travail ;

- des données qui sont dérivables d'autres données ne doivent pas faire l'objet d'une entrée, mais doivent être calculées automatiquement : si l'ordinateur connait la date du jour et a demandé la date de naissance de l'utilisateur, il ne faut plus qu'il lui demande son âge ;
- le nombre de caractères à entrer (ou de touches à enfoncer) doit être minimisé (par exemple, par l'utilisation d'abréviations : utilisation de "del" pour "delete", ...)

Une deuxième recommandation est de concevoir un écran avec beaucoup de clarté pour la facilité de l'utilisateur. Plus l'interface est clair, plus il est simple, moins il est susceptible d'amener des erreurs. Voici donc quelques conseils à suivre :

- toutes les données doivent apparaitre à l'écran, sauf les données confidentielles : une commande doit apparaitre à l'écran dans son entièreté (même si on n'a enfoncé qu'une touche pour la lancer) tandis qu'un mot de passe ne doit jamais s'afficher à l'écran au quel cas il perdrait son caractère confidentiel vu qu'il pourrait être vu par n'importe qui ;
- toutes les données entrées lors d'une même opération doivent être visibles tant que l'opération n'est pas terminée. Par exemple, si l'ordinateur demande d'entrer le nom et le prénom de quelqu'un, il faut laisser afficher le nom que l'utilisateur a entré pendant qu'il entre le prénom. Une fois l'opération de saisie de l'identité terminée, les deux

paramètres pourront être effacés ;

- un curseur doit indiquer l'endroit auquel le prochain caractère doit être tapé. Ce curseur doit se trouver en face du libellé du paramètre dont on demande la valeur

Exemple : nom : _

Contre-exemple : nom :

nom :

_

- il faut que les libellés soient protégés des interventions de l'utilisateur.

Une troisième recommandation concerne le langage de dialogue. Quelques conseils :

- il doit être adapté à l'application : les mots utilisés sont ceux qui sont effectivement utilisés pour la tâche ;
- les séquences de commandes doivent être simples ;
- il faut attendre la fin de l'entrée de la donnée pour imprimer la ou les sorties.

Voici encore des conseils généraux destinés à l'amélioration des entrées d'informations :

- les entrées doivent s'exécuter au rythme de l'utilisateur ;
- lorsque celui-ci a entré une commande, elle doit être confirmée en appuyant sur une touche de fin d'entrée ;

- les erreurs de formats et d'existence doivent être vérifiées après chaque entrée, avec obligation pour l'utilisateur de "ré-entrer" la donnée qui ne serait pas valide ;
- pour toutes les commandes, il faut un message de confirmation ou d'infirmation.

2.5.3 LES SORTIES

Un premier aspect très important en ce qui concerne les sorties est le temps de réponse. Celui-ci est le temps qui s'écoule entre une action de l'utilisateur et une réponse de l'ordinateur. Il varie très fort en fonction de la tâche mais il ne doit en aucun cas (sauf exception) dépasser 10 à 15 secondes. S'il dépassait ce temps, l'utilisateur perdrait le fil de ses idées en ce qui concerne les opérations à exécuter dans la réalisation de sa tâche. On considère que 2 secondes est un temps de réponse souhaitable. De plus, pour les mêmes actions, on doit obtenir les mêmes délais. Ainsi, lorsqu'on ré-exécute une action, on sait le temps qu'il faudra pour obtenir la réponse. Enfin, lorsque le temps de réponse est assez long, un message indiquant ce qu'est en train de faire le logiciel doit être présenté à l'utilisateur.

Deuxièmement, l'information présentée doit être pertinente : elle doit être lisible et elle doit être adaptée au niveau des utilisateurs. L'information ne doit pas être trop compliquée pour des utilisateurs non-expérimentés, car elle pourrait conduire à des erreurs dues à un manque de compréhension.

L'information affichée doit permettre un minimum d'interprétation de la part de celui qui s'en sert. De plus, l'utilisateur doit avoir à sa disposition une fonction d'aide (HELP) dont il pourra se servir en cas d'imcompréhension ou d'oubli.

Un troisième aspect important concerne le mode d'affichage des données. Un bon affichage permet d'éviter bon nombre d'erreurs et de faciliter l'emploi du logiciel. Les recommandations qui suivent guideront le concepteur d'interface dans sa recherche afin d'obtenir un affichage de qualité.

- la taille des caractères doit être telle que l'utilisateur sache lire les données sans forcer sa vue ;
- il faut rassembler les données qui font partie d'un même groupe d'information : si l'ordinateur demande des informations concernant deux opérations différentes sur le même écran, une partie de l'écran doit être réservée à TOUS les paramètres d'une des deux opérations, tandis qu'une autre partie doit être réservée à l'autre opération ;
- chaque groupe de données doit comporter un titre qui l'identifie ;
- l'utilisation de majuscules, de clignotement, etc. est souhaitée pour attirer l'attention de l'utilisateur : lorsqu'il fera une fausse manoeuvre, l'utilisateur en sera averti par un message d'erreur en majuscule accompagné d'un indicateur sonore de telle sorte qu'il sache qu'il a failli dans sa tâche et qu'il la recommence ;
- lorsque plusieurs écrans sont nécessaires à l'affichage d'un groupe de données, les différents écrans doivent être

numérotés ;

- une certaine organisation est nécessaire dans l'affichage des données : l'existence d'aires d'affichage spécialisées est souhaitée (les messages apparaîtront par exemple toujours au même endroit, les paramètres seront également demandés chaque fois dans une même partie de l'écran). La dernière ligne de l'écran sera par exemple réservée à l'affichage des messages à destination de l'utilisateur.

Une dernière recommandation est relative aux messages : ceux-ci doivent être fonction de l'expérience des utilisateurs. Pour des utilisateurs inexpérimentés, les messages devront être nombreux et les guider le plus possible dans l'exécution de leurs tâches. Par contre, des utilisateurs plus compétents n'apprécieraient guère de trop nombreux messages diminuant l'efficacité de leur travail mais se satisferaient plutôt d'un interface ne contenant que les messages strictement nécessaires. De plus, un message ne doit contenir que des informations utiles à l'utilisateur, ni plus ni moins : les données superflues pourraient embrouiller celui-ci. Enfin les messages doivent être clairs de telle manière que leur signification ne soit pas ambigüe.

2.6 INTERFACE D'UN COURRIER ELECTRONIQUE

Toutes les recommandations faites ci-dessus s'appliquent évidemment dans le cas d'un interface destiné à un logiciel de

courrier électronique. On peut aussi donner quelques autres conseils, plus spécifiques au courrier électronique, qui peuvent aider à la conception d'un interface de plus grande qualité, et dès lors, faciliter l'utilisation de ce logiciel. Ces conseils seront plus techniques, directement relatifs aux concepts du courrier électronique. Les voici :

- lorsque l'abonné au courrier électronique a reçu un document demandant une réponse, il est essentiel que, pour former celle-ci, il ait la possibilité d'avoir sous les yeux ET le document auquel il va répondre ET la réponse qu'il est en train de composer (figure 2.1) ;

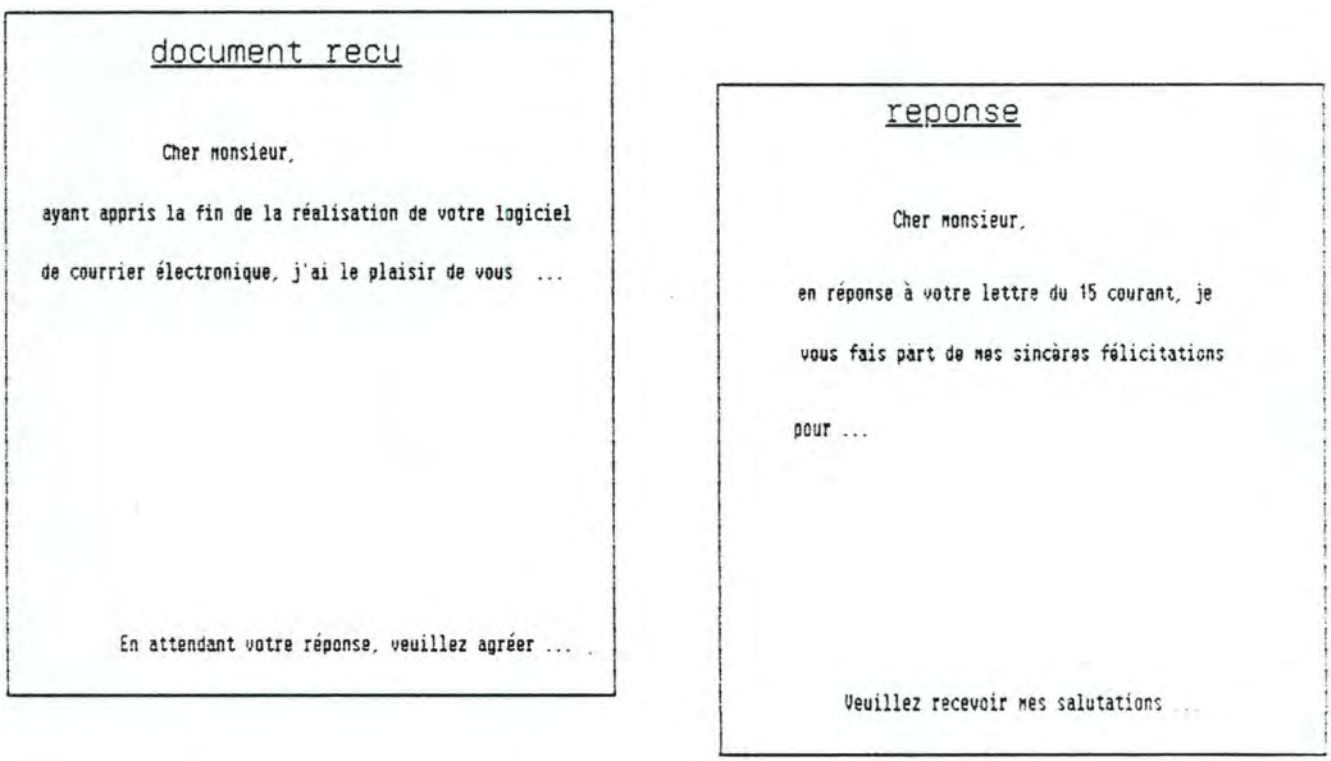


Figure 2.1: REPONSE APPARAISSANT A COTE DU DOCUMENT LA MOTIVANT

- lorsqu'il consulte une collection de documents, l'utilisateur doit pouvoir atteindre n'importe quel document dans un temps raisonnable : ceci nécessite des procédures d'accès rapide à ces documents ;
- lorsque l'utilisateur envoie un document à d'autres abonnés, il doit pouvoir obtenir la liste de ceux-ci et entrer leur nom avec la liste sous les yeux ;
- celui qui envoie un document à un autre abonné doit obtenir une indication comme quoi le document en question a bien été envoyé ;
- l'utilisateur doit pouvoir demander une impression des documents qu'il a reçus ou qu'il a composés lui-même.

2.7 CONCLUSION

Après avoir énoncé tous ces principes, tous ces conseils, on peut remarquer que plusieurs grands traits généraux se dégagent. La plupart des recommandations convergent vers trois grands points qui englobent plus ou moins tout ce qui a été dit :

- il faut différencier les interfaces pour les utilisateurs expérimentés des interfaces pour les utilisateurs non-expérimentés ;
- il faut faciliter au maximum la tâche des utilisateurs ;
- il faut faire en sorte que les possibilités de commettre des erreurs soient minimisées

Avant d'appliquer ces recommandations à la conception d'un interface pour un logiciel de courrier électronique (chapitre 4), nous allons au préalable analyser les outils fournis par le matériel sur lequel cet interface sera implanté.

3 ENVIRONNEMENT DE BASE

3.1 INTRODUCTION

Dans ce chapitre, nous allons décrire le contexte dans lequel ce mémoire a été réalisé. Cette description est nécessaire car ce contexte est assez différent de celui qui avait été employé lors de la première implémentation [mémoire de A. JOSIS et P. BERNARD, "Courrier électronique"].

Nous analyserons cet environnement sous deux points de vue : le hardware et le software. L'environnement hardware est constitué de la machine SUN, et l'environnement software du système des fenêtres (windowing).

3.2 ENVIRONNEMENT HARDWARE

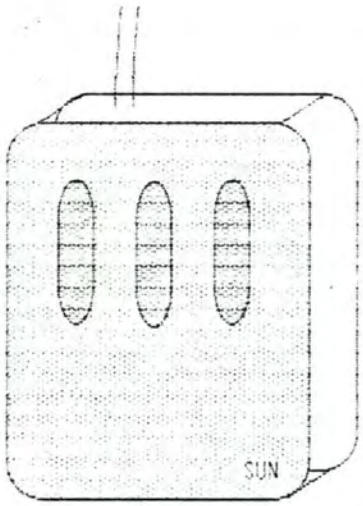
La machine que nous avons utilisée pour la réalisation de ce mémoire est un SUN (SUN Microsystems, Inc.). Le SUN a des caractéristiques techniques qui font qu'il est particulièrement bien adapté pour des programmes où l'interface est un des aspects les plus importants (par exemple pour un programme de courrier électronique).

En effet, l'écran du SUN a une résolution hors du commun

(1100 points de largeur pour 900 points de hauteur), et est ce qu'on appelle un "bit-mapped display" : une partie de la mémoire est réservée pour l'écran, et chaque point de l'écran peut être allumé ou éteint individuellement en positionnant le bit correspondant en mémoire. Ce principe, combiné à la haute résolution, permet d'afficher facilement beaucoup de choses en même temps à l'écran.

On utilise avec le SUN une souris sur laquelle se trouvent trois boutons. La souris est représentée à l'écran par un curseur (souvent une flèche). Quand on déplace la souris, le curseur se déplace de manière correspondante sur l'écran. Ceci permet de désigner un point bien précis de celui-ci, beaucoup plus aisément qu'avec les flèches de direction du clavier. De plus, le curseur reste à la même place tant qu'on ne touche pas à la souris, ce qui est un avantage par rapport à un "light pen" qui permet également de désigner facilement un point de l'écran. Ce principe de la souris permet notamment de choisir une commande parmi d'autres lorsqu'elles sont affichées à l'écran, en déplaçant le curseur (et donc la souris) sur la commande voulue et en appuyant à ce moment sur le bouton prévu à cet effet.

Le SUN dispose d'une mémoire centrale de 2 Mbytes et d'un disque dur de 48 ou de 72 Mbytes. Ce sont de grosses mémoires, mais elles sont nécessaires car les programmes avec un interface évolué en sont gros consommateurs (pour stocker des parties de l'écran qui ne sont plus momentanément visibles par exemple). Mais vu l'évolution du prix des mémoires, ceci ne peut plus être



souris du SUN

Figure 3.1: SOURIS DU SUN

considéré comme un handicap. De plus, le SUN peut être connecté à un réseau Ethernet.

3.3 ENVIRONNEMENT SOFTWARE

3.3.1 UTILISATION DES FENETRES

3.3.1.1 LES FENETRES

Le système d'exploitation du SUN possède deux modes de travail : le mode normal et le mode graphique. En mode normal, le SUN travaille comme un terminal tout à fait classique, avec un

écran de 24 lignes sur 80 colonnes. Cependant, les caractères sont particulièrement grands et lisibles. En mode graphique, le SUN travaille selon le principe des fenêtres. Il est donc obligatoire de passer au mode graphique pour pouvoir travailler avec le système des fenêtres. Ce passage se fait au moyen de la commande SUNTOOLS.

Vu les dimensions de l'écran, il est possible de diviser celui-ci en plusieurs morceaux rectangulaires. On les appelle des FENETRES. On peut alors associer à chacune de ces fenêtres une application, ou un "tool", qui consiste en un ou plusieurs processus UNIX [AT&T Bell Laboratories]. Un "tool" est une application complète avec un interface basé sur le système des fenêtres. On pourrait donc considérer une fenêtre comme un écran virtuel, dans la mesure où le "tool" s'exécute à l'intérieur de cette fenêtre, de la même manière qu'une application classique s'exécute à l'intérieur d'un écran classique.

Le rôle du "tool" est de contrôler la fenêtre, c-à-d gérer ses déplacements, ses modifications de taille, les données contenues dans cette fenêtre, mais également interpréter les événements qui sont dirigés vers cette fenêtre, c-à-d les actions effectuées avec la souris (lorsque le curseur est dans cette fenêtre) et les entrées provenant du clavier.

Il existe un certain nombre de "tools" standards, définis dans le système, comme "clocktool" qui donne l'heure, ou "shelltool" qui simule un terminal sur lequel s'exécute un shell

UNIX (un shell est un programme qui gère les entrées de commandes d'un système d'exploitation), et où il est donc possible d'exécuter n'importe quelle commande UNIX ou n'importe quel programme (voir la figure 3.2). Mais il est également possible de programmer soi-même ses propres "tools" (voir le point 3.3.2 Programmation des fenêtres).

Plusieurs fenêtres peuvent être présentes en même temps à l'écran, et dans chacune d'entre elles s'exécute une application qui est tout à fait indépendante des applications s'exécutant dans les autres fenêtres. Ces applications, qui sont en fait des processus UNIX différents, s'exécutent en parallèle, selon le principe de parallélisme entre processus du système UNIX. Toutefois, les entrées provenant du clavier ou de la souris ne sont dirigées que vers une seule fenêtre, celle où le curseur est présent.

En utilisant la souris, il est possible d'effectuer un certain nombre d'opérations sur les fenêtres. Il est notamment possible de les déplacer, et de les superposer. D'ailleurs, lorsqu'une fenêtre est créée (soit par une commande tapée dans une fenêtre "shelltool", soit par une commande venant d'un menu), elle se positionne par défaut au milieu de l'écran, sur les fenêtres qui seraient éventuellement déjà présentes à cet endroit. Il est à noter que si une fenêtre se trouve sur une partie d'une autre fenêtre, lorsqu'on déplace la première de telle manière qu'elle ne se trouve plus sur la deuxième, celle-ci est évidemment reconstruite pour retrouver son état normal. Il est également

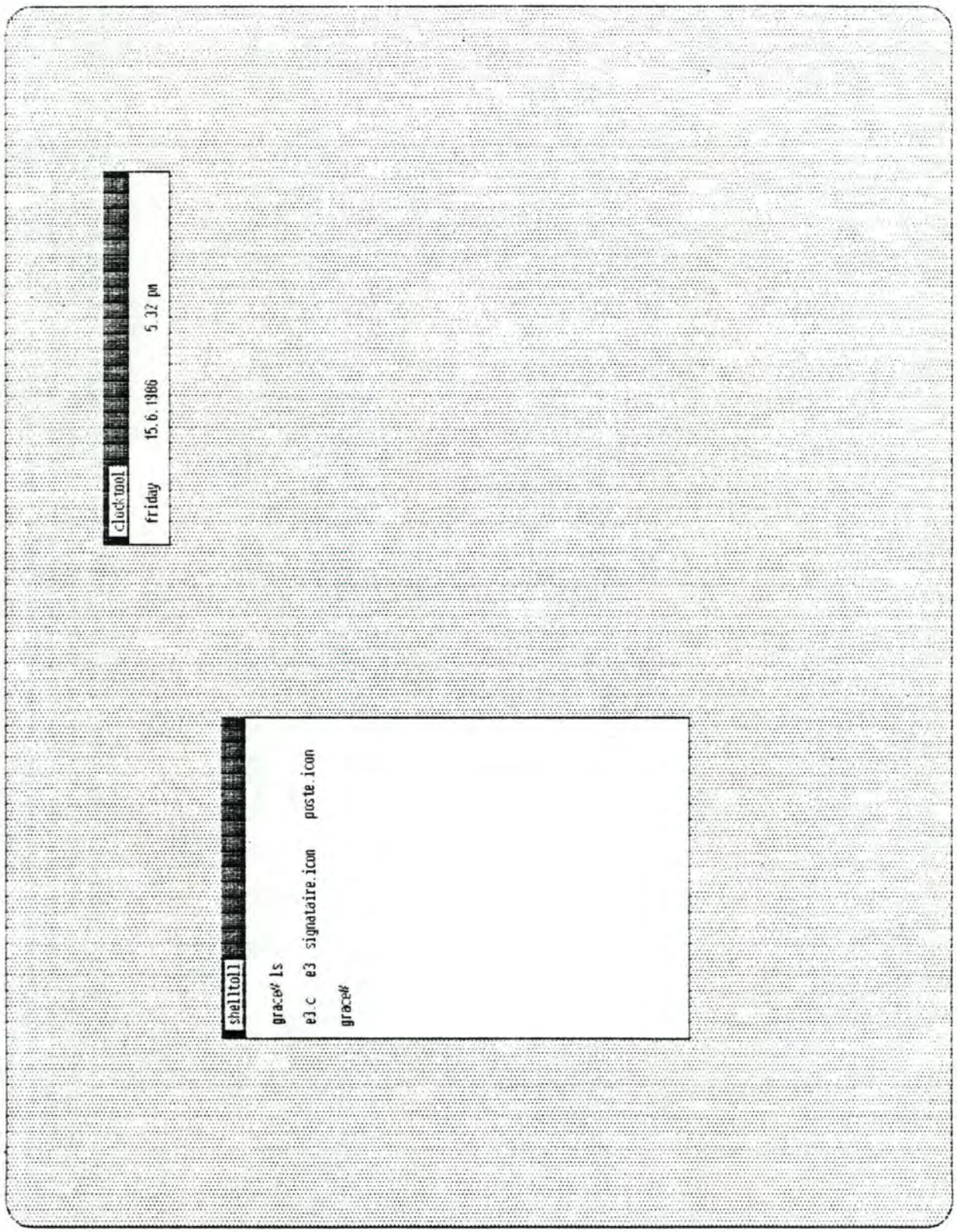


Figure 3.2: TOOLS STANDARDS SHELLTOOL ET CLOCKTOOL

possible à tout moment de ramener une fenêtre au dessus des autres, ou de la mettre en dessous des autres. Toutes ces opérations n'entravent en rien le déroulement des applications qui s'exécutent dans les fenêtres concernées. Par exemple, sur la figure 3.3, on a déplacé la fenêtre qui contient "clocktool" sur celle qui contient "shelltool". Mais les deux "tools" continuent à fonctionner tout à fait normalement, et si on redéplace "clocktool" pour le remettre à l'endroit ou il se trouvait sur la figure 3.2, "shelltool" sera reconstruite et redeviendra également comme sur la figure 3.2.

Il est également possible d'agrandir ou de rétrécir ces fenêtres. Si vous rétrécissez une fenêtre, vous perdez la partie que vous venez d'enlever. Toutefois, lors de l'utilisation suivante de cette fenêtre, le système tient compte de sa nouvelle taille.

Pour quitter un "tool", il faut utiliser soit la commande de sortie du "tool" (mais tous les tools n'en ont pas une), soit la commande QUIT du menu "tool-manager" (voir le point 3.3.1.4.2 Le tool-manager). Quand on a quitté un "tool", la fenêtre associée à ce "tool" disparait de l'écran.

3.3.1.2 LES SOUS-FENETRES

Il y a toujours dans une fenêtre une ou plusieurs sous-fenêtres. Les sous-fenêtres servent à séparer les différents types d'information qui se trouvent dans une fenêtre. Par

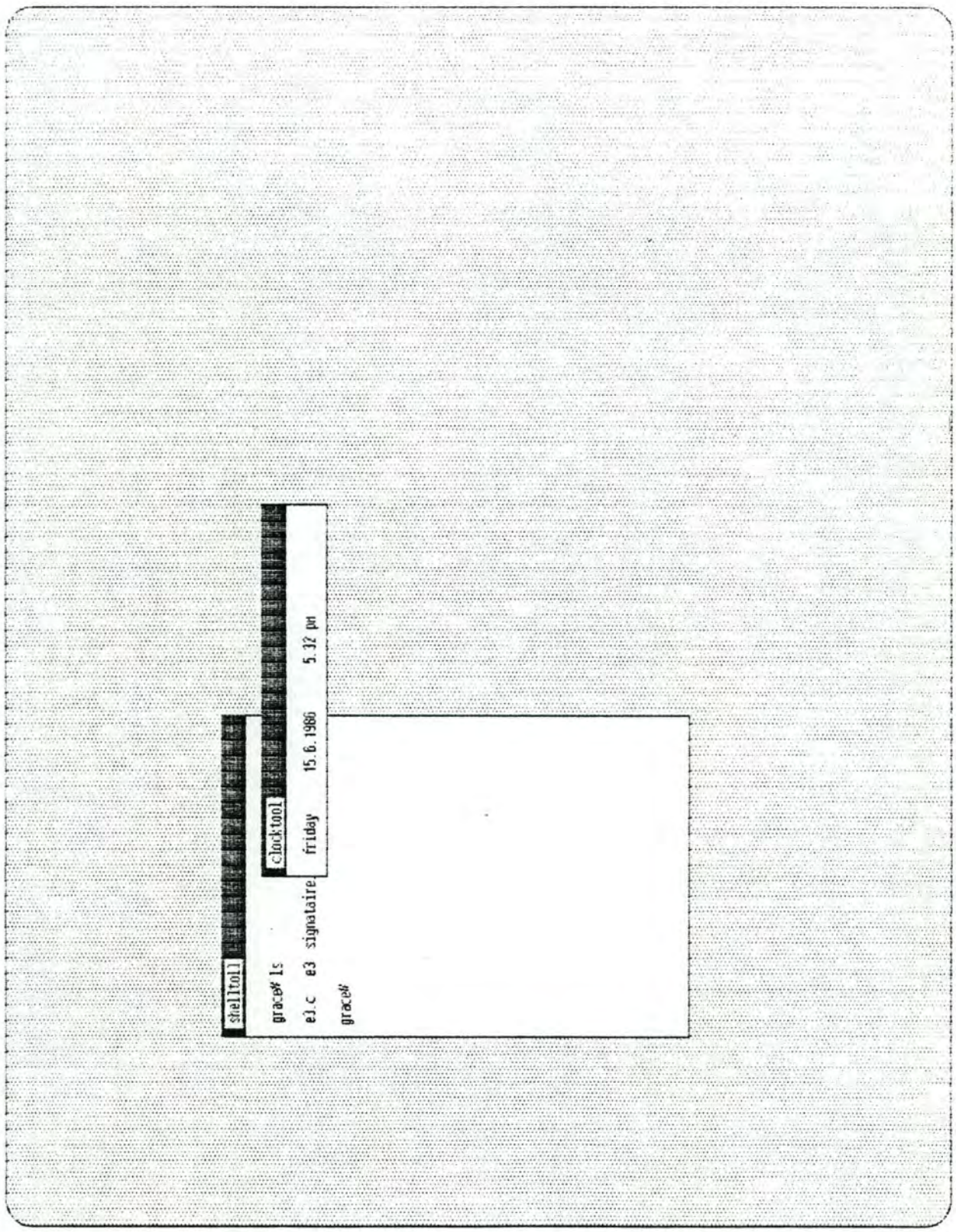


Figure 3.3: DEPLACEMENT DES FENETRES

exemple, on peut avoir une sous-fenêtre où apparaissent des messages, une sous-fenêtre où se trouve une série de commandes, et une sous-fenêtre où apparaissent les résultats découlant de ces commandes.

Certains types de sous-fenêtres sont définis dans le système :

- les sous-fenêtres de messages, où l'on peut imprimer des messages en utilisant une fonction qui a pour paramètres le nom de la sous-fenêtre et le message à imprimer ;
- les sous-fenêtres de panneaux de commandes (voir point 3.3.1.5 : les panneaux de commandes) ;
- les sous-fenêtres graphiques où des graphiques peuvent être dessinés ;
- les sous-fenêtres de simulation de terminal, où s'exécute un shell UNIX.

Ces sous-fenêtres sont gérées par le système, c-à-d par exemple que le système les reconstruit lui-même si elles ont été "endommagées" par une autre fenêtre qui est venu se superposer à elles. Mais il est possible également de définir ses propres sous-fenêtres quand les sous-fenêtres prédéfinies ne conviennent pas. Ceci implique que ces sous-fenêtres doivent contrairement aux sous-fenêtres prédéfinies être entièrement gérées par le programme utilisateur (ce point sera développé en détail au point 3.3.2 Programmation des fenêtres).

Contrairement aux fenêtres, les sous-fenêtres d'une même fenêtre ne peuvent pas être superposées. Elles peuvent simplement être mises une à coté de l'autre. Les sous-fenêtres bougent en même temps que la fenêtre qui les inclut, sans changer de place à l'intérieur même de celle-ci. Si la fenêtre est rétrécie ou agrandie, les sous-fenêtres sont dans la plupart des cas rétrécies ou agrandies proportionnellement à leur taille par rapport à la taille de la fenêtre.

Sur la figure 3.4, on voit un "tool" composé de deux sous-fenêtres : au dessus, une sous-fenêtre de simulation de terminal, et en dessous une sous-fenêtre où l'on peut dessiner des graphiques. Quand on tape dans la première un nom de programme de dessin graphique, il s'exécute dans la seconde.

3.3.1.3 LES ICONES

Un icône est un dessin de petite taille qui symbolise le "tool" associé à une fenêtre. Il existe donc un et un seul icône par fenêtre. Ils servent à remplacer les fenêtres sur l'écran, quand celles-ci ne sont plus utilisées temporairement.

En effet, il est possible de "fermer" une fenêtre, ce qui revient à la soustraire de l'écran et à la remplacer par son icône. Pour fermer une fenêtre, on utilise la commande CLOSE du menu obtenu sur le bord de cette fenêtre (voir le point 3.3.1.4.2 Le tool-manager). Une fois une fenêtre fermée, on ne voit

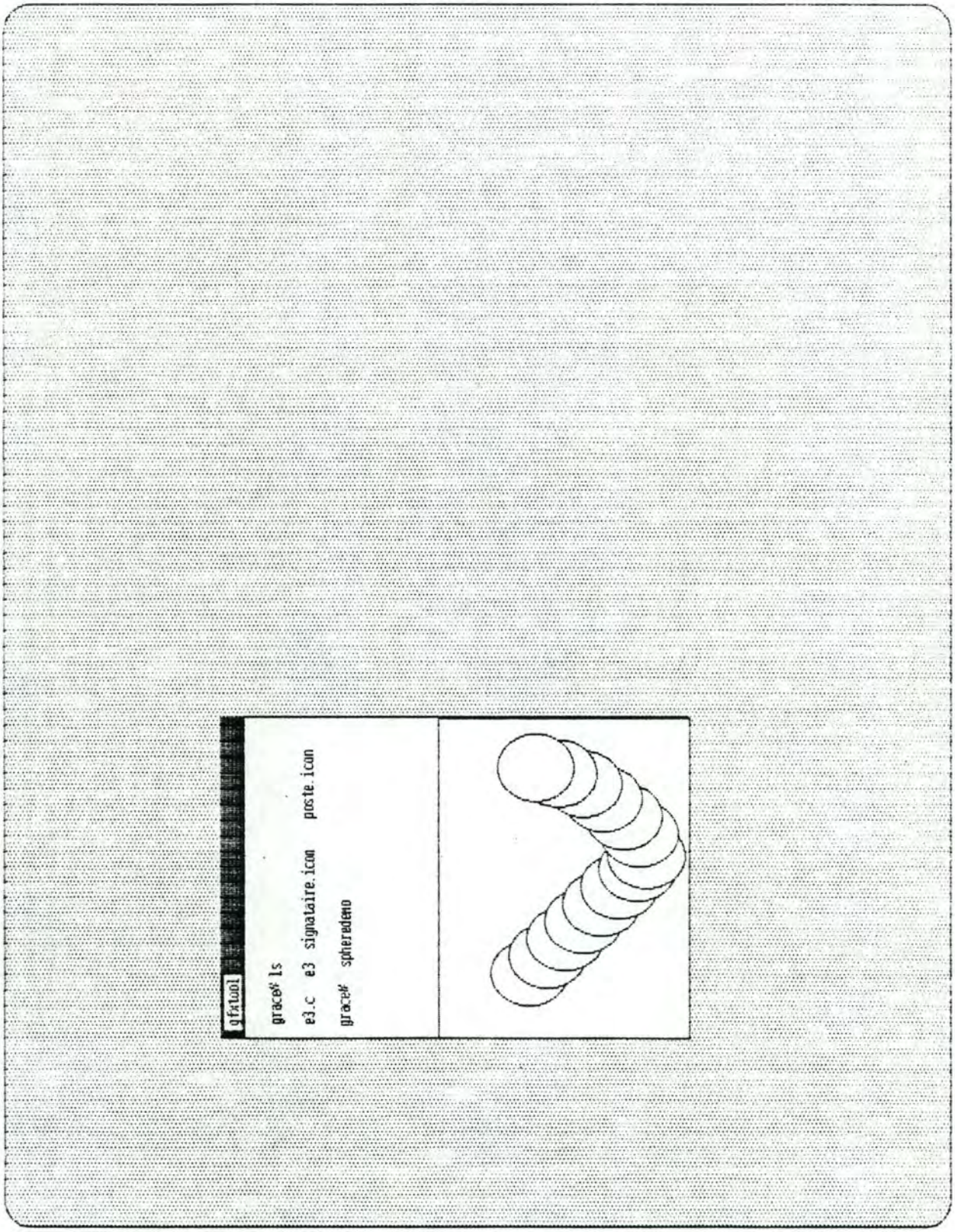


Figure 3.4: SOUS-FENETRES

évidemment plus son contenu, mais l'application qui tournait dans cette fenêtre continue à se dérouler normalement. Quand l'utilisateur le souhaite, il peut "réouvrir" une fenêtre en positionnant le curseur sur l'icône qui symbolise cette fenêtre et en utilisant la commande OPEN du menu qu'il obtient (voir le point 3.3.1.4.2 Le tool-manager). La fenêtre réapparait alors à l'endroit où elle se trouvait quand elle a été fermée.

En général, les icônes vont se placer au bas de l'écran. Mais le programmeur d'un "tool" peut placer l'icône de celui-ci où il le désire sur l'écran. Les icônes peuvent au même titre que les fenêtres se trouver sur une fenêtre, sous une fenêtre, où même sur ou sous un autre icône. Mais il est possible de les déplacer de la même manière que les fenêtres.

L'interêt des icônes est de clarifier l'écran, notamment quand il y a sur celui-ci de nombreuses fenêtres qui ne servent temporairement à rien. Il est alors préférable de les fermer, puis de les réouvrir quand le besoin s'en fait sentir.

Par exemple, sur la figure 4.5, on voit que la fenêtre "shelltool" a été fermée, et on voit son icône en bas de l'écran.

3.3.1.4 LES MENUS

Il existe plusieurs méthodes pour envoyer des commandes à un "tool". Il est par exemple possible de demander à l'utilisateur de taper sa commande dans une des sous-fenêtres du "tool", mais

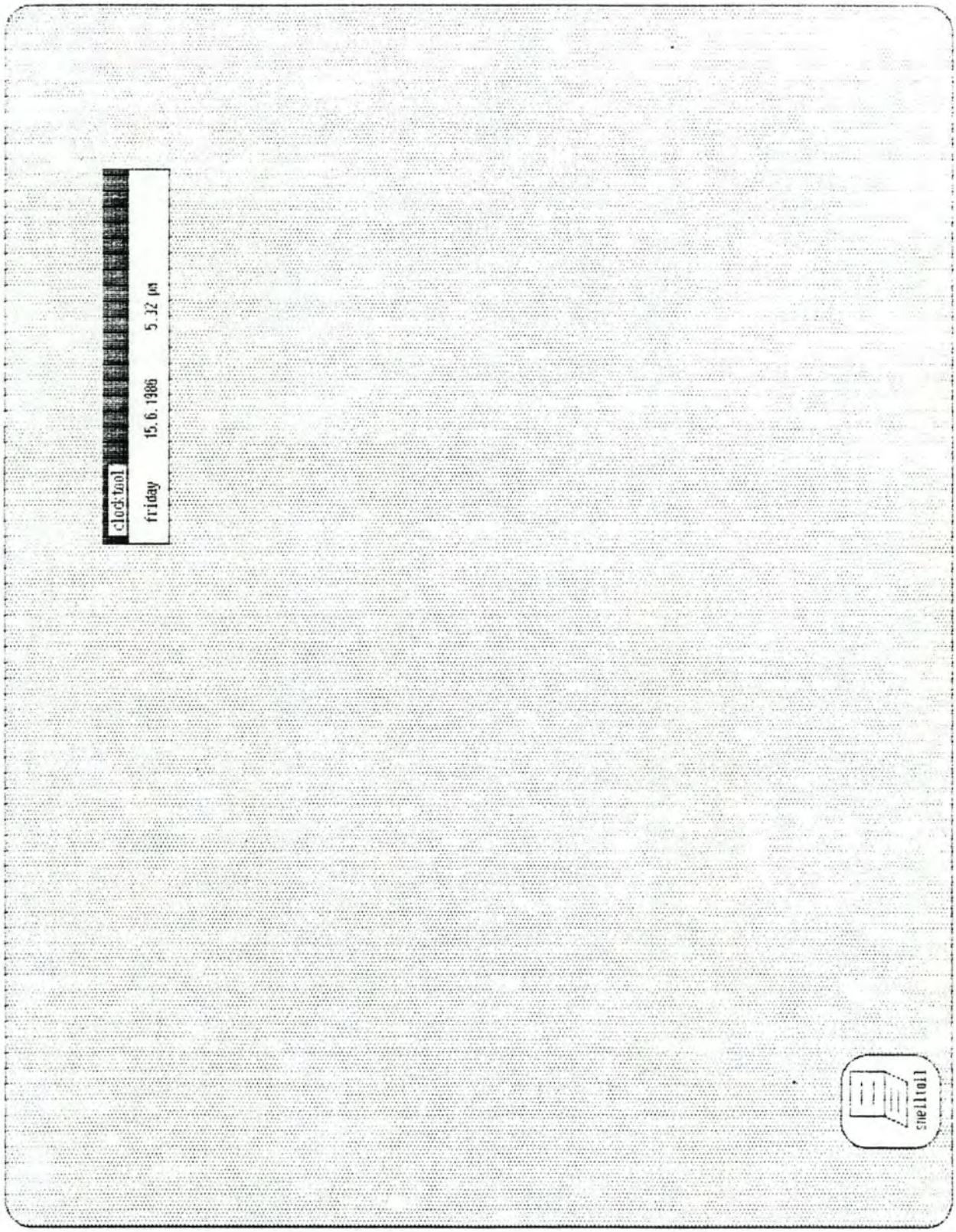


Figure 3.5: ICONES

cette méthode requiert de l'utilisateur qu'il se souvienne du nom des commandes, et qu'il les tape au clavier.

Cette méthode (qui est celle utilisée par le shell UNIX par exemple) n'est pas très satisfaisante (comme vu au point 2.4.3 Qualités d'un bon interface - la concision). C'est pourquoi on utilise d'autres méthodes, et notamment celle des menus et des panneaux de commandes.

Un menu n'est présent à l'écran que lorsque l'utilisateur en a besoin (ceci pour économiser la place sur l'écran). Pour appeler un menu, il faut en règle générale appuyer sur le bouton droit de la souris. Le menu apparait alors et reste à l'écran tant que l'on garde enfoncé le bouton droit.

Un menu est composé d'au moins deux cases. La première contient le titre du menu et est en inverse (c-à-d en blanc sur fond noir). Là où les autres cases contiennent les commandes du menu (qui seront appelées dans la suite "commandes-menu"). Pour choisir une commande-menu, il suffit de déplacer le curseur dans la case qui contient la commande désirée tout en gardant le bouton droit enfoncé. Quand le curseur se trouve dans une case, celle-ci s'affiche en inverse. La commande qui sera exécutée sera celle qui est en inverse (c-à-d celle qui est inclue dans la case où le curseur se trouve) quand l'utilisateur lâche le bouton droit (voir figure 3.6). Si le curseur ne se trouve dans aucune case quand le bouton droit est relaché, aucune commande ne sera exécutée. Dans tous les cas, le menu disparait lorsque l'utilisateur lâche le

bouton droit.

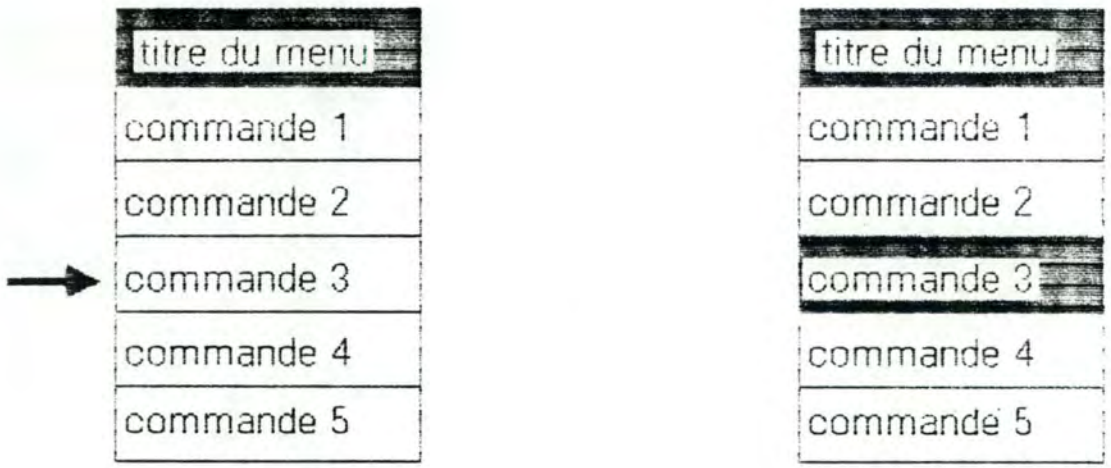


Figure 3.6: MENUS

Il est bien entendu possible d'avoir plusieurs menus disponibles à un moment donné sur l'écran. Le menu qui apparait lorsqu'on appuye sur le bouton droit de la souris est différent selon l'endroit où se trouve le curseur au moment où on appelle le menu. En fait, il existe trois types de menu qui peuvent apparaitre :

- le "root-manager" (figure 3.7) ;
- le "tool-manager" (figure 3.8) ;
- les menus propres aux "tools".

3.3.1.4.1 LE ROOT-MANAGER

Le "root-manager" est le menu qui apparait quand le curseur se trouve sur le fond gris de l'écran, c-à-d sur aucune fenêtre ou icône. Ce menu se compose notamment d'une commande pour revenir au mode "terminal classique" du SUN, d'une commande pour créer une fenêtre "shelltool", et une commande pour réafficher l'écran après une perturbation (par exemple un message du système d'exploitation).

Quand l'utilisateur rentre dans le mode graphique du SUN (par la commande SUNTOOLS), l'écran est complètement gris et vide. Le seul moyen pour pouvoir travailler est de créer une fenêtre "shelltool" en appelant le menu "root-manager", puis de taper les commandes voulues dans cette fenêtre. Il est également possible de définir un fichier dans lequel se trouvent le ou les "tools" que l'utilisateur veut voir présents sur l'écran quand il exécute la commande SUNTOOLS (c-à-d quand il rentre dans le mode graphique du SUN).

3.3.1.4.2 LE TOOL-MANAGER

Le "tool-manager" est le menu qui apparait quand le curseur se trouve sur le bord d'une fenêtre (à ce moment la flèche qui représente le curseur se transforme en un cercle) ou sur un icône.

C'est un menu commun à tous les "tools" (même ceux programmés

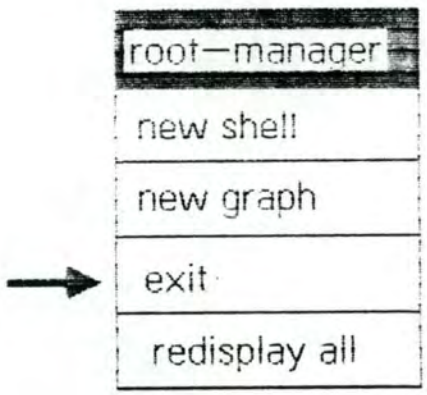


Figure 3.7: MENU ROOT-MANAGER

par un utilisateur), qui permet de faire les opérations de maintenance sur le "tool". Les différentes commandes du "tool-manager" sont :

- CLOSE (passer à la forme iconique de la fenêtre) ou OPEN (passer de la forme iconique à la forme normale) selon que l'on est sur le bord d'une fenêtre ou sur un icône ;
- MOVE (déplacer la fenêtre ou l'icône) ;
- STRETCH (agrandir ou rétrécir une fenêtre) ;
- EXPOSE (mettre la fenêtre ou l'icône au dessus des autres fenêtres et icônes) ;
- HIDE (mettre la fenêtre ou l'icône en dessous des autres fenêtres et icônes) ;
- REDISPLAY (réafficher la fenêtre) ;
- QUIT (terminer le "tool").

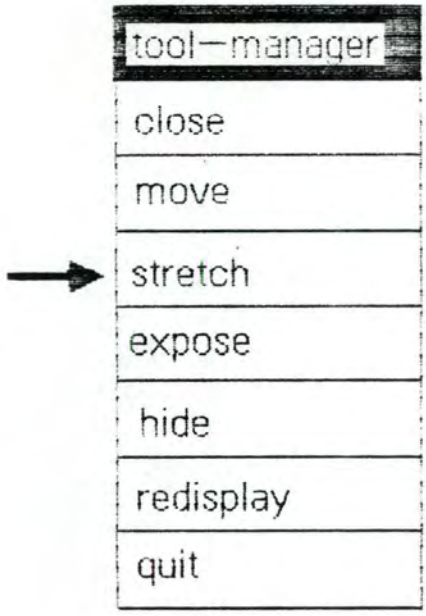


Figure 3.8: MENU TOOL-MANAGER

3.3.1.4.3 **LES MENUS PROPRES AUX TOOLS**

Les menus propres aux "tools" apparaissent quand le curseur est à l'intérieur d'une sous-fenêtre d'un "tool" et qu'il existe un menu défini pour cette sous-fenêtre. Les commandes de ces menus dépendent évidemment du "tool" et de la sous-fenêtre. Chaque sous-fenêtre d'un "tool" peut avoir un menu différent.

C'est dans cette catégorie qu'apparaissent les menus que l'on peut définir dans un programme.

3.3.1.5 LES PANNEAUX DE COMMANDE

Le panneau de commande est une alternative au menu. Il consiste en une série d'items qui restent en permanence affichés à l'écran (figure 3.9). Ces items sont représentés soit par un texte, soit par un graphique. On sélectionne un item de la même manière qu'avec les menus : il suffit de déplacer le curseur sur l'item désiré et d'appuyer sur le bouton de gauche.

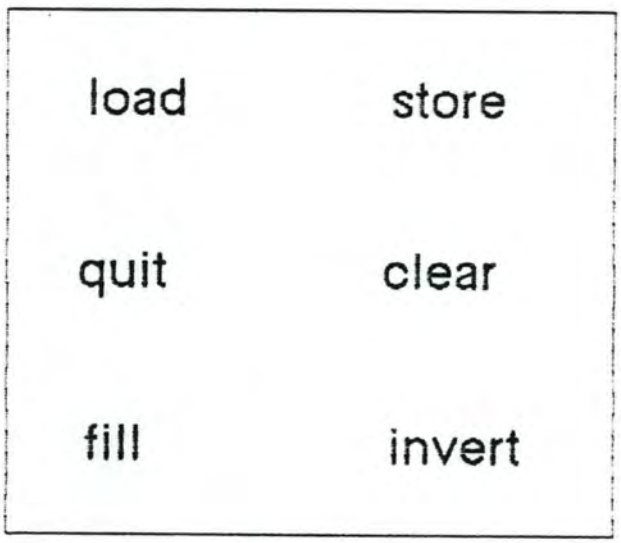


Figure 3.9: PANNEAU DE COMMANDE

Cependant, l'avantage des panneaux de commande sur les menus est que l'on peut avoir plusieurs types d'items différents dans un panneau. Parmi ceux-ci, il y a les "boutons" (pour appeler une commande), les choix (une liste de paramètres ou d'options), les booléens (un choix à deux états), les textes (pour remplir des variables comme par exemple des noms de fichier), ...

De plus, les panneaux et les menus peuvent être utilisés en même temps. On peut par exemple mettre les commandes fréquemment utilisées dans un panneau, tandis que les commandes moins utilisées seront dans un menu.

3.3.1.6 DEMARRAGE D'UN TOOL

Pour démarrer un "tool", il est nécessaire de se trouver dans le mode graphique (pour rappel, on y accède par la commande SUNTOOLS). Puis il faut soit taper le nom du "tool" dans une fenêtre "shelltool", soit avoir mentionné le nom de ce "tool" dans un fichier où sont définis les "tools" qui sont automatiquement démarrés lors de la commande SUNTOOLS.

3.3.2 PROGRAMMATION DES FENETRES

3.3.2.1 OUTILS DISPONIBLES

Pour programmer des "tools", le programmeur a à sa disposition des librairies qui contiennent des procédures (en fait des fonctions écrites en langage C) mais aussi des déclarations de structures de données. Ces librairies sont divisées en trois niveaux (voir figure 3.10). Le niveau 2 se sert des procédures définies dans le niveau 1, et le niveau 3 se sert des deux niveaux précédents. Le niveau 3 est donc le niveau supérieur, et les "tools" font appel le plus souvent à ce niveau. Mais le

programmeur peut se servir des fonctions et des structures définies aux trois niveaux. Il est également possible de modifier ou de remplacer tout ou une partie de ces niveaux pour par exemple intégrer un autre appareil de pointage (pointing device) que la souris. Il en résulte une grande flexibilité du système.

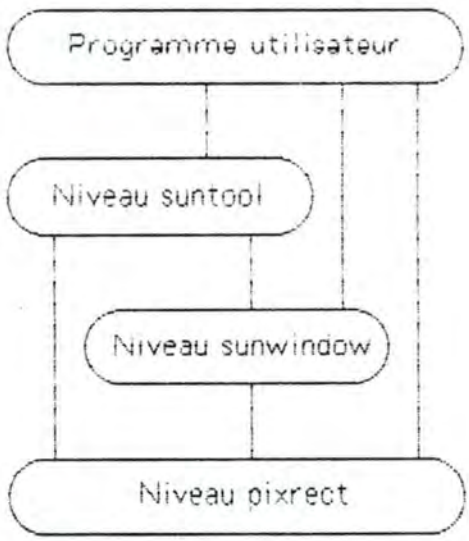


Figure 3.10: NIVEAUX DES LIBRAIRIES DU SYSTEME DES FENETRES

3.3.2.1.1 LE NIVEAU PIXRECT

Le niveau "pixrect" offre des procédures sur des pixels groupés en rectangles appellés "pixrect". La notion de rectangle de pixels est très vague (mais c'est volontaire). Il peut s'agir d'un caractère, de l'image d'un curseur, d'un icône, d'une fenêtre ou même de l'écran en entier. Ce qu'il est important de souligner, c'est que les procédures sont fort générales pour, par exemple, pouvoir travailler de la même façon sur des objets qui sont déjà à l'écran ou qui sont en mémoire.

3.3.2.1.2 LE NIVEAU SUNWINDOW

A partir des procédures définies au niveau "pixrect", le niveau "sunwindow" construit un système de fenêtres qui peuvent se superposer. Il maintient une trace des zones de fenêtres qui sont cachées et gère le fait qu'il ne faut rien afficher dans une partie de fenêtre qui est cachée.

Le niveau "sunwindow" gère les déplacements et dessine le curseur à l'écran. Il définit les boutons de la souris, et gère les entrées au clavier. Chaque action est estampillée pour pouvoir calculer le temps écoulé entre deux actions.

De plus, il existe des procédures à ce niveau pour déterminer la taille des fenêtres, pour écrire des rectangles, des vecteurs, du texte ... dans une fenêtre sans devoir s'occuper du fait de savoir si cette fenêtre n'est pas recouverte par une autre. Il y a également moyen de déterminer la position du curseur, de le déplacer, et de changer son image à l'écran.

3.3.2.1.3 LE NIVEAU SUNTOOL

Le niveau "suntool", en collaboration avec le programme utilisateur, intègre les fenêtres définies au niveau précédent dans un "tool". On obtient alors une application complète avec un cadre, une bande en inverse (blanc sur noir) sur la partie supérieure du cadre dans laquelle s'inscrit le nom du "tool", un

icône par défaut, une disposition par défaut des sous-fenêtres dans le "tool", et le menu "tool-manager".

Il existe également à ce niveau des procédures pour créer des menus, des sous-fenêtres et des icônes.

3.3.2.2 TECHNIQUES DE PROGRAMMATION DES TOOLS

Un "tool" est un programme ordinaire qui utilise des fonctions définies dans les trois niveaux (vus au point 3.3.2.1 Outils disponibles). Cependant, les entrées/sorties (souris, clavier et écran) d'un "tool" sont différentes d'un programme ordinaire. De plus, un "tool" a toujours la même structure générale, quel qu'il soit. C'est cette structure qui est développée dans ce chapitre et qui est reprise schématiquement sur la figure 3.11.

3.3.2.2.1 DEFINITION DE LA FENETRE

La fenêtre est définie grâce à une fonction du niveau "suntool". Cette fonction a de nombreux paramètres qui permettent de modifier les valeurs par défaut définies dans le système, comme par exemple la taille de la fenêtre, sa position sur l'écran, le texte dans la bande de titre, l'icône qui symbolise cette fenêtre, et le fait de savoir si la fenêtre apparait ouverte ou fermée.

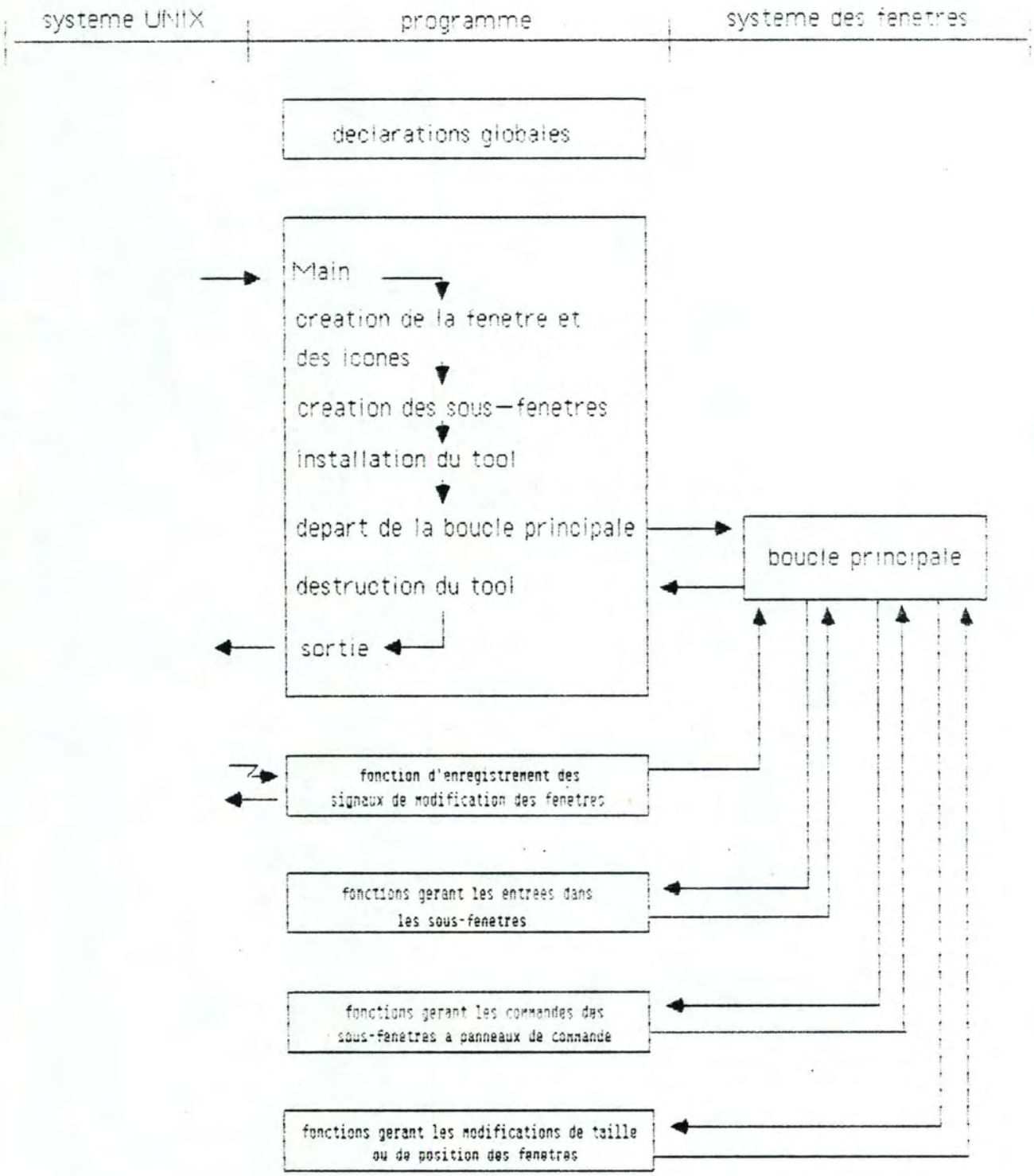


Figure 3.11: STRUCTURE GENERALE D'UN TOOL

3.3.2.2.2 DEFINITION DES SOUS-FENETRES

Une fois la fenêtre définie, il est nécessaire de définir la ou les sous-fenêtres qui constituent le "tool". Ceci se fait de manière analogue à celle utilisée pour la définition de la fenêtre, c-à-d que l'on emploie une fonction du niveau "suntool" qui a plusieurs paramètres dont notamment la position de la sous-fenêtre à l'intérieur de la fenêtre, et sa taille.

Il existe un certain nombre de sous-fenêtres standards définies par le système (voir point 3.3.1.2 Les sous-fenêtres). A chaque type de ces sous-fenêtres correspond une fonction (par exemple la fonction "msgsw_createtoolsubwindow" crée une sous-fenêtre de messages). Pour les sous-fenêtres définies par l'utilisateur, il existe une fonction générale (tool_createsubwindow) qui définit une sous-fenêtre sans s'occuper de ce que l'on fera dedans.

3.3.2.2.3 INSTALLATION DE LA FENETRE

Une fois la fenêtre et ses sous-fenêtres définies, il reste à les installer à l'écran. Ceci se fait grâce à la fonction "tool_install". Cette fonction ajoute la fenêtre à la base de données des fenêtres gérée par le système, et envoie un signal UNIX (SIGWINCH) qui indique que la fenêtre doit être dessinée à l'écran (voir point 3.3.2.2.6 Reponse aux modifications de taille et de position).

3.3.2.2.4 REPONSE AUX ENTREES

Le principe fondamental de la programmation d'un "tool" est le suivant : un "tool" est ce qu'on appelle "event-driven", c-à-d conduit par les événements. Au départ, le "tool" ne fait rien, il est dans un état d'attente. En fait, il attend un événement, qui peut être une action avec la souris (un déplacement ou le fait d'appuyer sur un des boutons), une touche du clavier ou un timer qui se termine. Dès qu'un événement se produit, le "tool" y répond (en fonction de ce qu'il y a dans le programme), puis retourne à son état d'attente de l'événement suivant.

Tout ceci se passe à l'intérieur d'une fonction du niveau "suntool", "tool_select". Cette fonction attend un événement, puis en fonction du type de cet événement et de la sous-fenêtre où se trouve le curseur au moment où l'événement se produit, elle passe le contrôle à la fonction qui s'occupe de ce type d'événement dans cette sous-fenêtre (ces fonctions sont écrites par l'utilisateur et elles déterminent ce que fait réellement le "tool"). "tool_select" boucle jusqu'à la terminaison du "tool" (voir point 3.3.2.2.7 Terminaison du tool).

Le fait que le programme est conduit par les événements entraine une programmation très différente de la programmation classique. Par exemple, si on veut lire un mot tapé par l'utilisateur dans une des sous-fenêtres du "tool", on doit lire ce mot caractère par caractère. En fait, chaque caractère tapé par l'utilisateur est un événement. On fait donc appel à la

fonction de lecture pour chaque caractère tapé, et pas une seule fois pour lire tous les caractères un après l'autre. On quitte donc la fonction de lecture après chaque caractère, ce qui entraine qu'on doit conserver quelque part ce qui a déjà été tapé du mot, et le nombre de caractères déjà tapés. Le problème se corse quand on doit lire plusieurs mots dans une fenêtre, car il faut retenir quel mot l'utilisateur est en train de taper. Enfin, ce n'est pas parce que l'utilisateur tape un caractère que celui-ci s'affiche à l'écran. Mais encore faut-il l'imprimer au bon endroit, car la fonction qui affiche un caractère à l'écran a notamment comme paramètre la position dans la sous-fenêtre du caractère à afficher. Il faut donc retenir également la position du caractère suivant à afficher dans la sous-fenêtre.

3.3.2.2.5 REPONSE AUX MODIFICATIONS DE TAILLE ET DE POSITION

L'utilisateur peut quand il le désire modifier la taille ou la position d'une fenêtre. Ceci implique que la fenêtre doit être redessinée, mais il est impossible de prévoir dans le programme quand redessiner la fenêtre, puisque cet événement est asynchrone par rapport à l'exécution du programme. C'est pourquoi le système utilise un signal UNIX appelé SIGWINCH pour signaler au programme qu'il faut redessiner la fenêtre. Lorsque ce signal est reçu par le "tool", un indicateur est positionné de telle manière que lorsque le programme reviendra à "tool_select" (c-à-d en attente de l'événement suivant), avant de traiter cet événement, la fenêtre sera redessinée.

Pour redessiner la fenêtre, il est nécessaire d'avoir pour chaque sous-fenêtre du "tool" une fonction qui s'occupe de redessiner sa sous-fenêtre. Cette fonction est prédéfinie pour les sous-fenêtres standards, mais le programmeur doit les définir lui-même pour les sous-fenêtres qui ne sont pas standards.

3.3.2.2.6 TERMINAISON DU TOOL

Quand l'utilisateur en a terminé avec le "tool" (commande-menu QUIT ou une des commandes du "tool" qui sert à en sortir), le programme sort de la fonction "tool_select". Pour terminer correctement, il suffit d'appeler la fonction "tool_destroy" qui enlève la fenêtre de l'écran et qui libère les ressources allouées au "tool".

3.4 CONCLUSION

Grâce à ses spécifications techniques et ses logiciels, le SUN est une machine qui est très bien adaptée pour les programmes où l'interface tient un rôle important. L'écran est très grand, une souris est reliée à la console, et les logiciels de base permettent de programmer des "tools", c-à-d des applications utilisant les principes du "windowing" (fenêtres qui se recouvrent, menus appelés par la souris, icônes ...).

Tout ceci fait qu'il est possible de réaliser sur le SUN des programmes avec un interface répondant aux qualités qui sont définies dans le chapitre 2.

Dans le chapitre suivant, nous allons montrer comment nous avons modifié l'interface du programme de départ, en utilisant les caractéristiques propres au SUN, pour essayer d'améliorer cet interface.

4 INTERFACE DEVELOPPE

4.1 INTRODUCTION

Le but du mémoire étant d'améliorer l'interface d'un logiciel, il est nécessaire de pouvoir dire en quoi nous l'avons amélioré. Nous allons donc d'abord décrire les principes généraux de l'interface de départ. Ensuite, nous donnerons les caractéristiques du nôtre. Nous regarderons s'il a les qualités d'un bon interface, qualités définies dans le chapitre 2.

Après cela, nous donnerons la structure générale de l'écran, suivie de la description des sous-fenêtres. Et pour terminer, on verra l'exemple de fonctionnement de la réception d'un document.

4.2 PROBLEMES DE L'INTERFACE DE DEPART

Vu l'ordinateur utilisé, la première implantation du courrier électronique à la base du nôtre n'a pu développer un interface de qualité.

Cet interface se fait par menus (voir 2.5.1 : le dialogue proprement dit). Ne disposant que d'un écran de taille modeste, et de surcroit ne pouvant diviser celui-ci en plusieurs parties, les auteurs du mémoire de départ ont utilisé le principe d'une

hiérarchie de menus. Ceci implique plusieurs constatations :

- l'utilisateur avait à l'écran un sous-menu et il devait se rappeler de quel menu supérieur il provenait;
- il ne pouvait exécuter deux tâches en parallèle;
- il possédait très peu d'informations à la fois sous les yeux.

On a aussi remarqué que les menus comportaient parfois des choix de commandes assez longues, ce qui est néfaste pour la simplicité d'utilisation.

Un gros problème que nous avons remarqué également est la difficulté de créer un document. Pour ce faire, l'utilisateur devait soit terminer l'exécution du programme et se servir de l'éditeur disponible sur l'ordinateur (ce qui n'était pas très pratique), soit se servir d'un éditeur assez primitif créé par les réalisateurs du programme (ce qui n'était pas très performant).

Ayant remarqué ces problèmes et le fait que leur interface ne respectait pas beaucoup les qualités d'un bon interface énoncées au chapitre 2, nous avons essayé d'en concevoir un qui résoudrait ces problèmes et surtout qui respecterait les qualités requises.

4.3 CARACTERISTIQUES DE L'INTERFACE AMELIORE

4.3.1 ECRAN STATIQUE

La structure de l'écran a été définie une fois pour toutes. Une telle structure invariable permettra de faciliter la tâche de l'utilisateur qui sera vite habitué à la disposition des informations sur l'écran. Nous avons établi une subdivision en sous-fenêtre qui sera la même durant l'entièreté de l'exécution du programme (voir figure 4.1). Chaque sous-fenêtre possède donc une place et une fonction qui lui sont propres. Cette caractéristique aboutit au fait que les mêmes opérations se fassent toujours aux mêmes endroits et que les mêmes événements apparaissent toujours à la même place.

4.3.2 EDITEUR INTERNE AU PROGRAMME

Lorsqu'il veut taper un document, l'utilisateur peut se servir de l'éditeur VI à l'intérieur du programme, sans devoir sortir de celui-ci. Cet avantage est dû au fait qu'il existe des sous-fenêtres de simulation de terminal (voir 3.3.1.2 : les sous-fenêtres) dans lesquelles on peut exécuter des commandes de unix. Une commande est spécialement prévue à cet effet. Ceci est un très gros avantage par rapport à l'interface de départ : la création d'un document est nettement plus rapide, plus simple et plus performante.

4.3.3 SIMPLICITE DES COMMANDES

Lorsque nous avons étudié le programme de départ, nous avons remarqué que, dans les menus offerts à l'utilisateur, les actions étaient souvent les mêmes, mais s'exécutaient sur des objets différents. Il y avait, par exemple, "consulter la poubelle", "consulter les archives", "mettre les documents en attente de traitement", "mettre les documents en poubelle", "imprimer" (c'est-à-dire faire un transfert vers l'imprimante), etc.

Ayant ainsi observé toutes les commandes des menus généraux (ceux qui sont au sommet de la hiérarchie des menus, juste après le menu de départ), nous avons remarqué que nous pouvions scinder chacune des commandes en deux parties : une action, et un objet sur lequel s'exécute cette action. Nous avons donc scindé leurs commandes, et nous avons obtenu deux actions ("CONSULTER" et "TRANSFERER") et treize objets (voir description de la sous-fenêtre des icônes). A ces deux actions allant de pair avec un objet, nous avons du en ajouter deux autres plus particulières qui sont des commandes en elles-mêmes : "EDITER" et "QUITTER". Plutôt que d'avoir beaucoup de longues commandes, nous avons deux listes : une liste d'actions (CONSULTER, TRANSFERER, EDITER, QUITTER) et une liste d'objets représentés par des icônes (BOITE-IN, BOTTIN, POUBELLE, ARCHIVES, DOCUMENTS EN ATTENTE, IMPRIMANTE, BOITE-OUT, SIGNATAIRE, POSTE, MOT DE PASSE, REPONSES RECUES, DOCUMENT PERE, ADMINISTRATION). Pour exécuter une commande, il suffit donc de sélectionner une action et un objet.

4.3.4 TACHES EXECUTEES EN PARALLELE

Une grande caractéristique de cet interface est que plusieurs tâches peuvent être exécutées en parallèle. Ceci est dû au fait que l'écran est divisé en sous-fenêtres qui ont chacune une fonction bien spécifique. De ce fait, on peut commencer l'exécution d'une tâche dans une sous-fenêtre, changer de sous-fenêtre en déplaçant le curseur dans cette dernière, et commencer une nouvelle tâche tandis que la première continue à s'exécuter.

4.4 STRUCTURE GENERALE DE L'ECRAN

La fenêtre générale est divisée en six sous-fenêtres qui ont chacune une fonction qui leur est propre (voir figure 4.1) :

- la sous-fenêtre des messages
- la sous-fenêtre des paramètres
- la sous-fenêtre des documents
- la sous-fenêtre des résumés
- la sous-fenêtre des commandes
- la sous-fenêtre des icônes

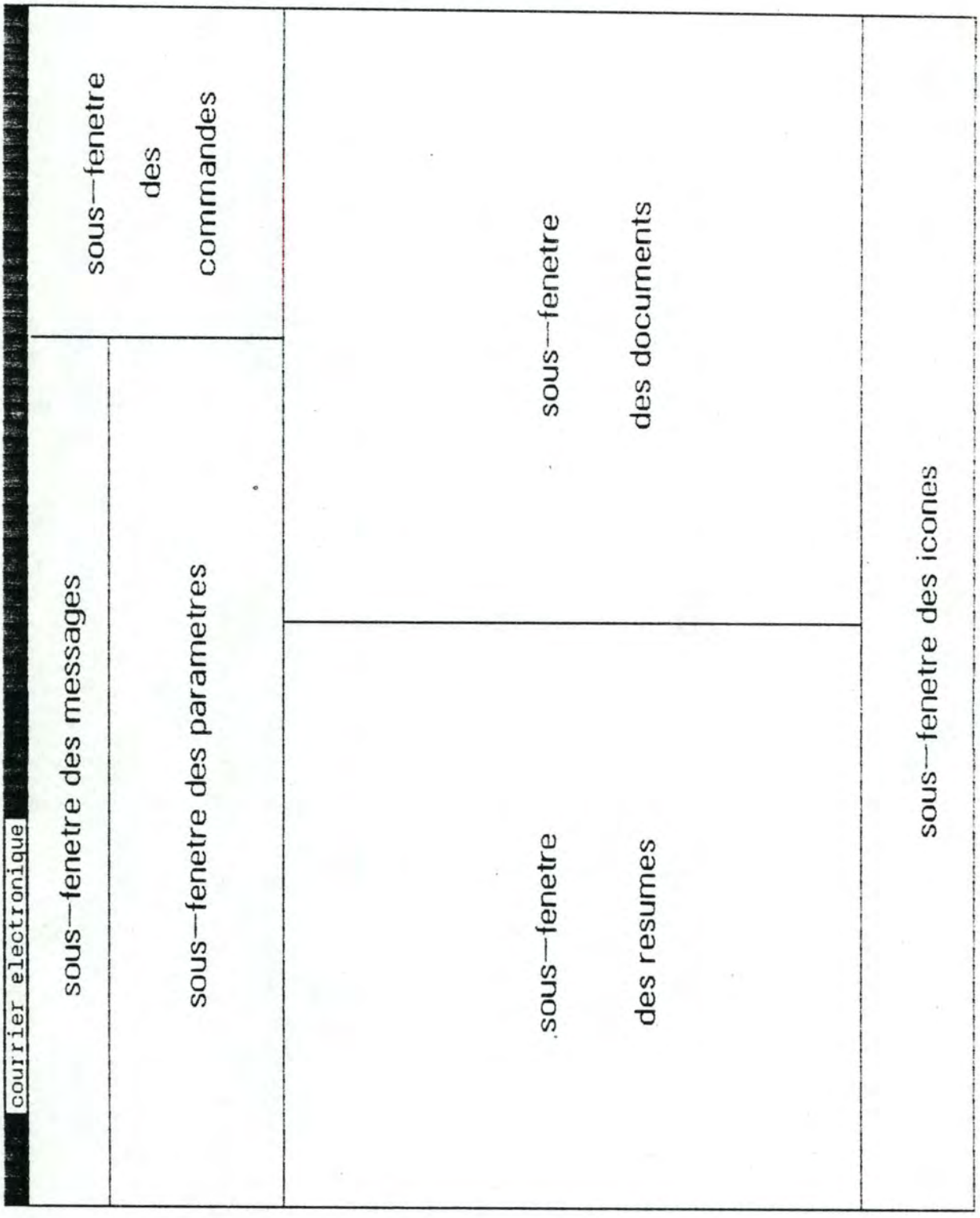


Figure 4.1: STRUCTURE GENERALE DE L'ECRAN

4.5 DESCRIPTION DES SOUS-FENETRES

4.5.1 SOUS-FENETRE DES MESSAGES

La sous-fenêtre des messages (figure 4.2) est utilisée, comme l'indique son nom, pour fournir à l'utilisateur les messages qui lui sont destinés . Cette sous-fenêtre est subdivisée en deux parties qui correspondent à deux types de messages qui peuvent survenir (figure 4.3) :

- il y a tout d'abord les messages qui donnent des indications quant à la situation dans le programme. Il est évident que l'utilisateur peut, à certains moments, ne plus se rappeler à quoi correspondent les documents se trouvant dans les deux sous-fenêtres prévues à cet effet . Le message "CONSULTATION POUBELLE" lui indiquera par exemple qu'il est en train de consulter soit les documents en poubelle, soit les résumés de ces documents . Ce type de messages se trouvera dans la partie droite de la sous-fenêtre;

- le deuxième type concerne les messages relatifs au déroulement des opérations que l'utilisateur est en train d'effectuer : messages d'erreur, de confirmation, d'avortement, etc. Le message "LISTE CREEE EXISTANT DEJA" le renseignera par exemple sur le fait que la liste qu'il veut créer existe déjà et qu'il ne peut pas, par conséquent, lui donner un tel nom .

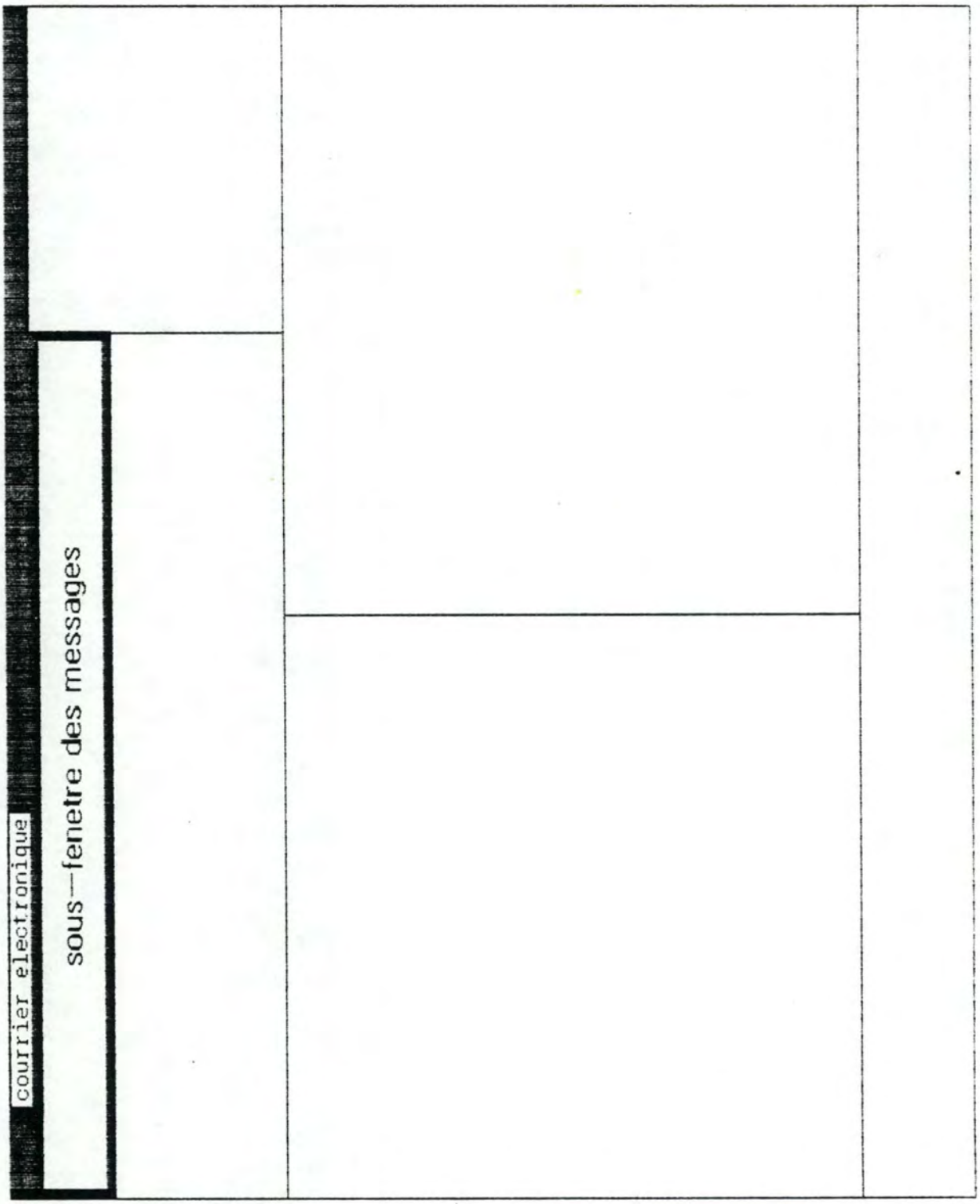


Figure 4.2: SOUS-FENETRES DES MESSAGES

	consultation poubelle
transfert effectue	transfert boite out
transfert annule	transfert boite out
liste creee existant deja	consultation bottin

Figure 4.3: EXEMPLE D'UTILISATION DE LA SOUS-FENETRE DES MESSAGES

Pour attirer l'attention de l'utilisateur, l'apparition des messages à l'écran est précédée d'un clignotement de cette sous-fenêtre.

4.5.2 SOUS-FENETRE DES PARAMETRES

Certaines fonctions ayant besoin d'information venant de l'utilisateur pour pouvoir s'exécuter, une sous-fenêtre a été créée à cet effet . La sous-fenêtre des paramètres (figure 4.4) est dès lors utilisée pour demander à l'utilisateur une ou plusieurs valeurs de paramètres qui sont nécessaires pour exécuter une opération. Lors de la création d'une nouvelle liste d'abonnés par exemple, c'est dans cette fenêtre que le nom de la liste créée et les noms des abonnés la formant, sont demandés .

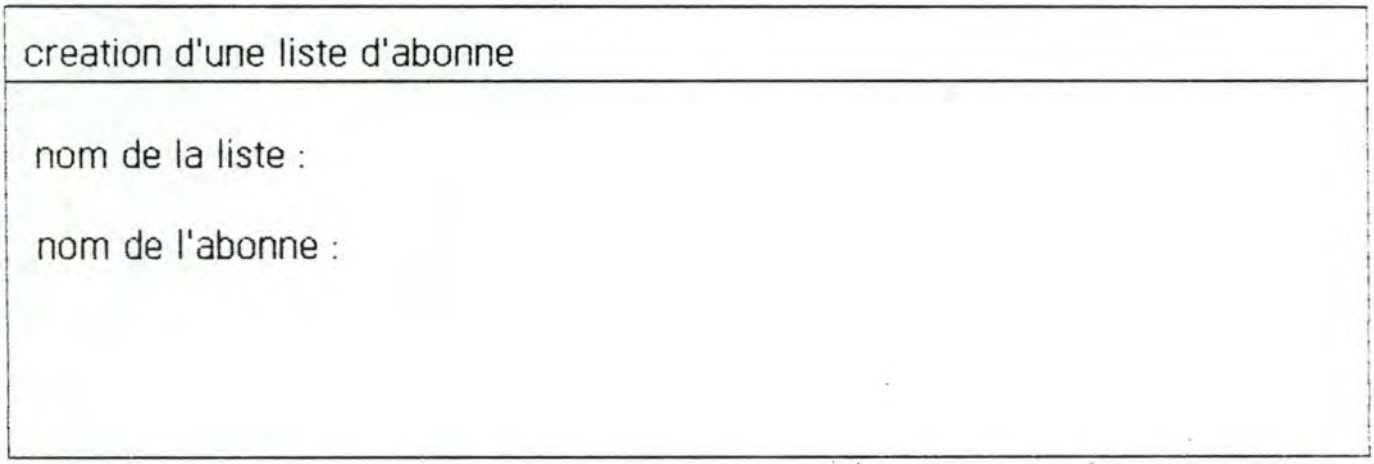


Figure 4.4: EXEMPLE D'UTILISATION DE LA SOUS-FENETRE DES PARAMETRES

Cette sous-fenêtre détermine fortement les possibilités qu'à l'utilisateur d'exécuter plusieurs fonctions en parallèle . En effet, étant donné qu'il n'existe qu'un endroit pour demander les paramètres, toute exécution d'une fonction qui requiert cet endroit, est prohibée si une autre fonction, activée plus tôt, se sert de cette sous-fenêtre des paramètres, et ceci jusqu'à ce que

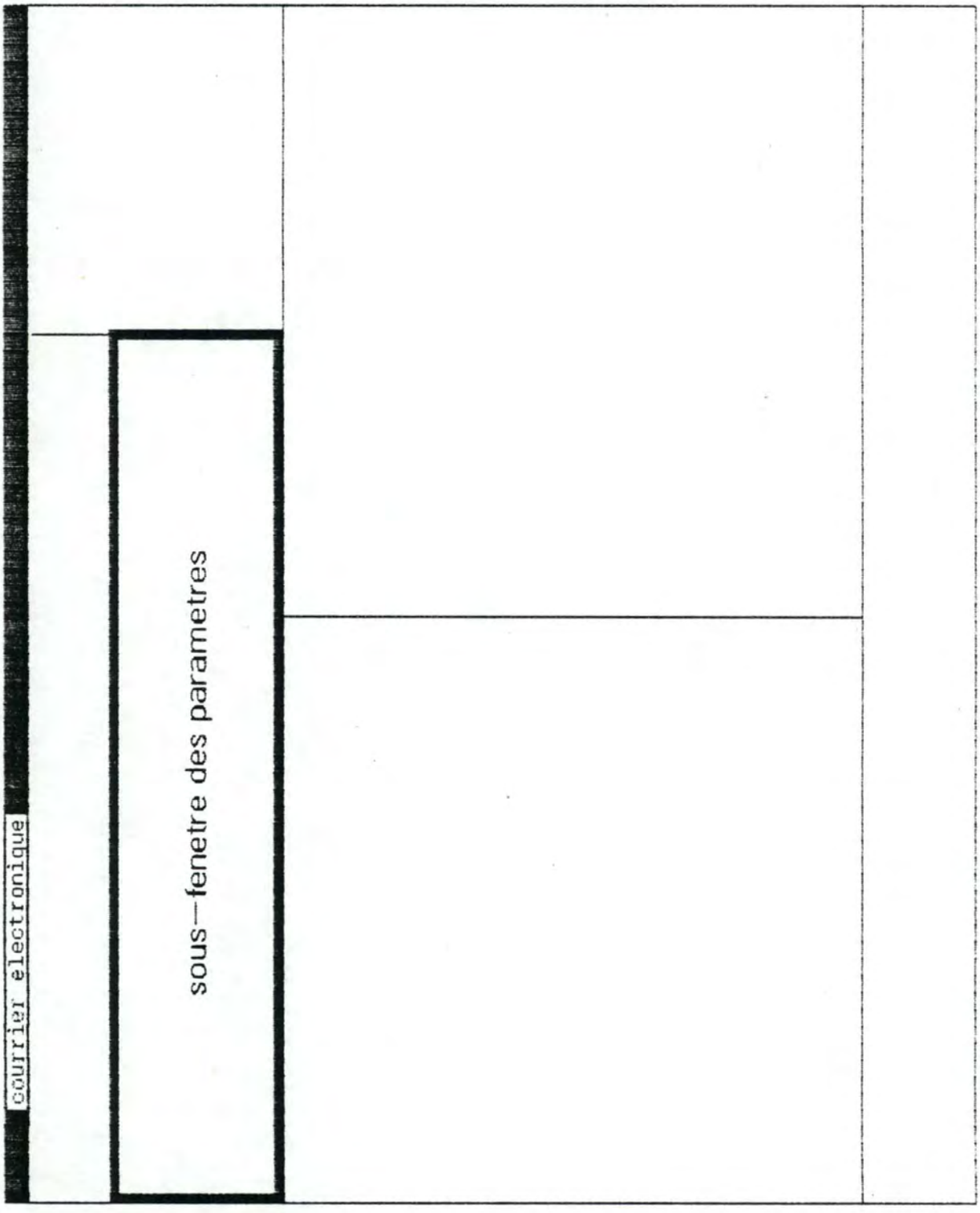


Figure 4.5: SOUS-FENETRE DES PARAMETRES

celle-ci soit libérée soit par la terminaison soit par l'avortement de la fonction qui s'en sert.

Prenons un exemple.

Supposons qu'un abonné soit en train d'effectuer un transfert de document à un autre abonné. Il utilise dès lors la sous-fenêtre des paramètres, dans laquelle lui sont demandés les paramètres d'envoi d'un document. Se rendant compte à ce moment qu'il envoie souvent des documents à un même groupe d'abonnés, il décide, tant qu'il y pense, de créer une nouvelle liste d'abonnés. Une telle opération de création requiert aussi l'utilisation de la sous-fenêtre des paramètres. Elle lui sera donc interdite tant que l'opération d'envoi ne sera pas terminée.

Les paramètres correspondant à une même fonction apparaissent tous en même temps dans la sous-fenêtre. Ils restent visibles à l'utilisateur au moins jusqu'à ce qu'il ait donné une valeur à l'ensemble de ceux-ci. En effet, l'utilisateur peut vouloir donner une valeur à un paramètre en fonction de la valeur qu'il a donné aux précédents. Il est donc intéressant de les avoir sous les yeux.

Une fois que l'utilisateur a introduit tous les paramètres demandés, un message apparait dans la sous-fenêtre des messages, qui lui indique quelle fonction il vient de terminer (par exemple après avoir introduit tous les paramètres nécessaires à l'envoi d'un document, le message "TRANSFERT BOITE OUT EFFECTUE" apparaitra dans la sous-fenêtre des messages) .

Néanmoins, ce qui a été tapé dans la sous-fenêtre des paramètres y restera jusqu'à ce qu'on ait besoin à nouveau de cette sous-fenêtre pour une autre fonction . Ceci dans le but de garder le plus d'informations possible à l'écran au sujet de ce qui s'exécute ou de ce qui vient de s'exécuter.

4.5.3 SOUS-FENETRE DES DOCUMENTS

La sous-fenêtre des documents (figure 4.6) est utilisée lors des consultations pour appeler à l'écran les documents correspondant à la collection choisie. Si, par exemple, l'utilisateur demande la consultation des documents de la poubelle en allant, à l'aide de la souris, placer le curseur successivement sur la commande CONSULTER et sur l'icône POUBELLE, le premier document de la poubelle apparaitra dans la sous-fenêtre des documents . Un menu permettra alors de passer d'un document à l'autre dans la collection, et d'effectuer d'autres opérations spécifiques à la collection dans laquelle on se trouve . Il n'y aura cependant jamais plus d'un seul document à la fois présent dans cette sous-fenêtre .

Prenons un exemple .

Lorsqu'on demande une consultation des documents de la poubelle, le premier document de cette collection est affiché dans la sous-fenêtre. A ce moment et au moyen de la souris, on peut obtenir un menu dans lequel se trouvent des fonctions telles que l'accès au document suivant, l'accès au document précédent, la

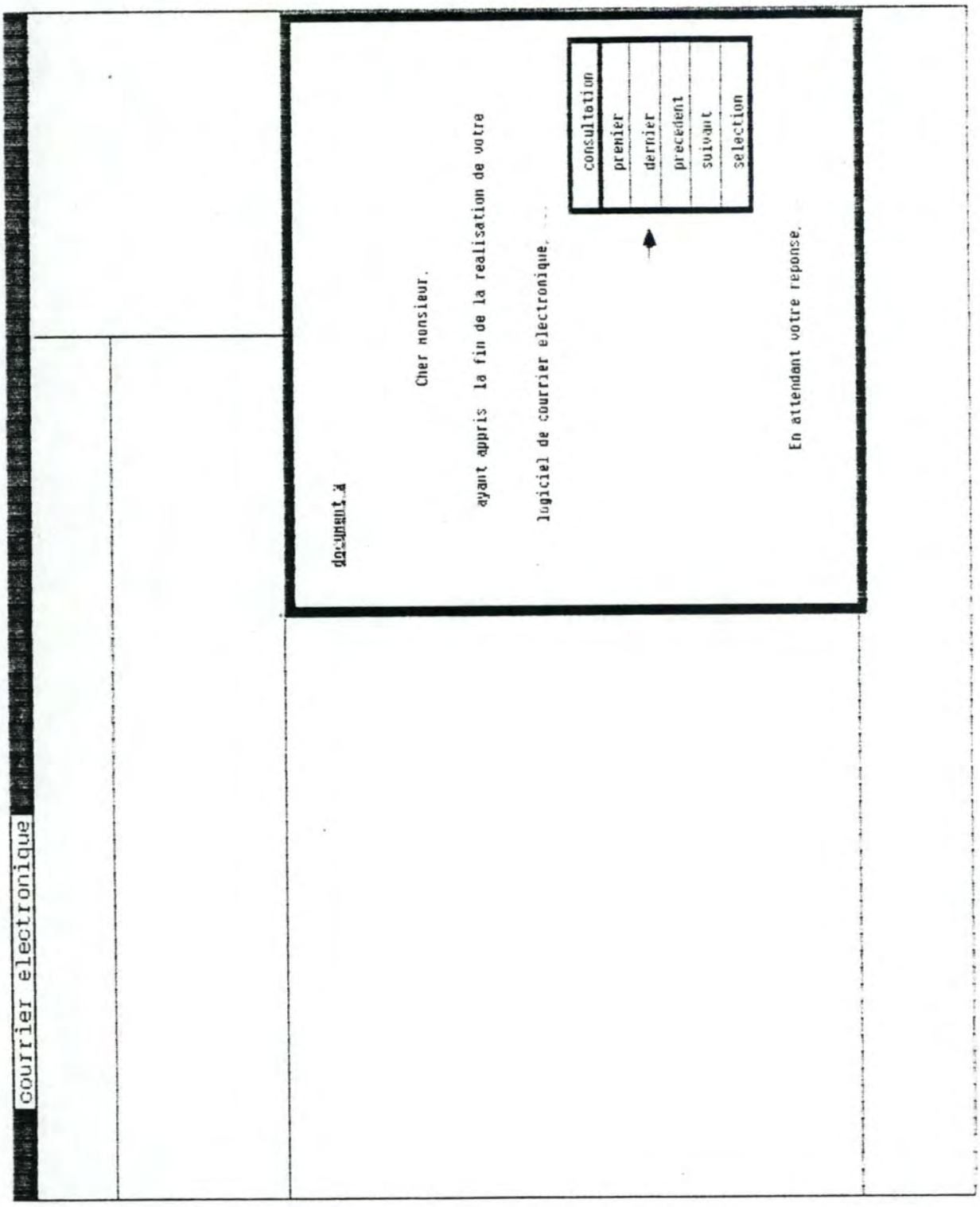


Figure 4.6: SOUS-FENETRE DES DOCUMENTS

sélection d'un document, etc. Si l'utilisateur choisit par exemple de voir le document suivant, le premier sera effacé de la sous-fenêtre et sera remplacé par le deuxième.

Cette sous-fenêtre est aussi utilisée pour la consultation du bottin, lequel n'est pas une collection de documents . Lorsque l'utilisateur exécute une telle consultation, la liste des abonnés, la liste des listes d'abonnés et les abonnés de chacune de ces listes peuvent être obtenus dans cette sous-fenêtre.

4.5.4 SOUS-FENETRE DES RESUMES

La sous-fenêtre des résumés (figure 4.7) est utilisée lors des consultations pour imprimer à l'écran les résumés des documents correspondant à la collection choisie . Comme pour les documents, un menu permettra de passer d'un résumé à l'autre, et d'effectuer dessus d'autres opérations dépendant de la collection dans laquelle l'utilisateur se trouve .

Avec la sous-fenêtre des documents et celle des résumés, nous avons donc, dans deux sous-fenêtres adjacentes, les documents d'une collection et leurs résumés . De plus, on peut accéder aux résumés d'une manière indépendante à celle dont on accède aux documents : on peut très bien avoir à l'écran le premier document de la collection dans la sous-fenêtre des documents, tandis que dans la sous-fenêtre des résumés, on a le résumé du troisième document de cette collection. Le résumé à l'écran n'est donc pas nécessairement celui du document qui s'y trouve également .

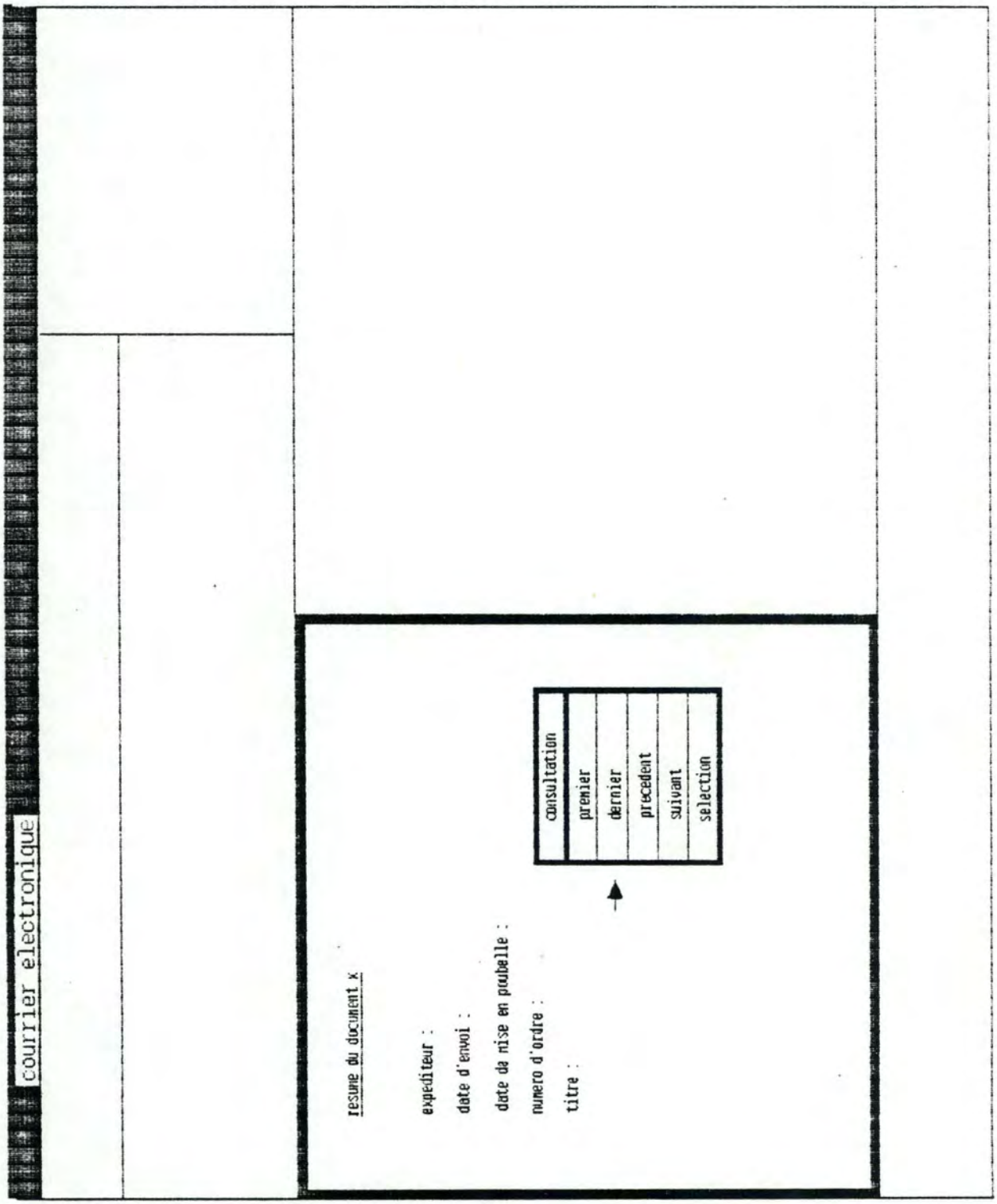


Figure 4.7: SOUS-FENETRE DES RESUMES

C'est également à l'endroit de cette fenêtre que l'éditeur apparait lorsque l'utilisateur le demande (voir la commande EDITER dans le paragraphe sur la sous-fenêtre des commandes) .

4.5.5 SOUS-FENETRE DE COMMANDES

La sous-fenêtre de commandes (figure 4.8) est composée de quatre commandes :

- CONSULTER
- EDITER
- TRANSFERER
- QUITTER

Ces quatre commandes peuvent être séparées en deux catégories :

- les commandes pour lesquelles il est nécessaire de définir l'objet sur lequel elles portent, c-à-d un icône : il s'agit des commandes CONSULTER et TRANSFERER ;
- les commandes pour lesquelles il n'est pas nécessaire de définir l'objet sur lequel elles portent : il s'agit des commandes EDITER et QUITTER .

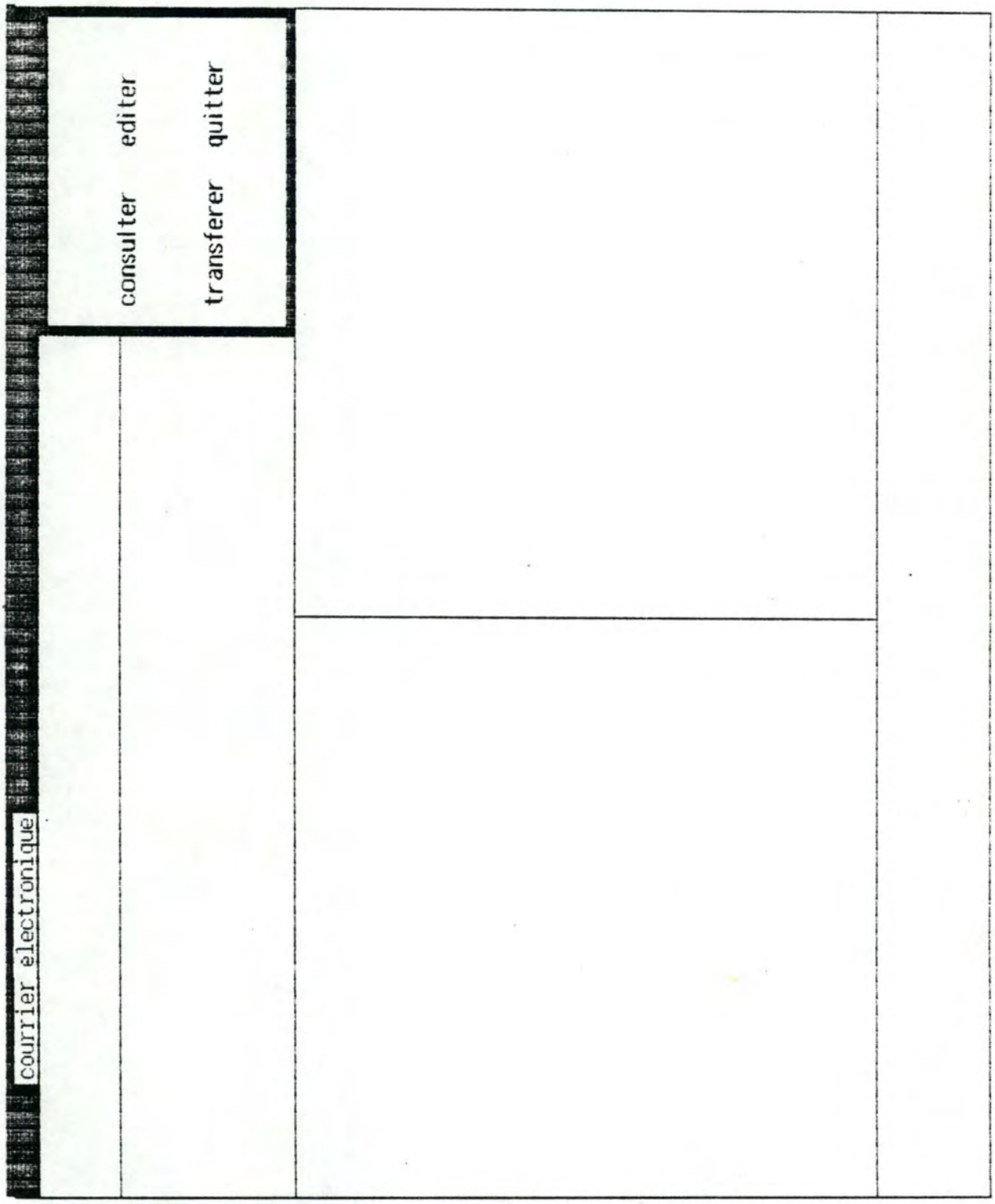


Figure 4.8: SOUS-FENETRE DES COMMANDES

4.5.5.1 LES COMMANDES PORTANT SUR UN ICONE

4.5.5.1.1 CONSULTER

La commande CONSULTER permet de faire la consultation d'une des collections de documents symbolisées par les icônes . Pour ce faire, il suffit de sélectionner la commande CONSULTER puis de sélectionner l'icône désiré .

4.5.5.1.2 TRANSFERER

La commande TRANSFERER est utilisée de la même manière que la commande CONSULTER . Il faut sélectionner cette commande, puis sélectionner l'icône sur lequel elle porte . Cependant, cette commande n'a de sens que si un document a été sélectionné auparavant . Si cela a été fait, c'est le document qui a été sélectionné qui est transféré de la collection dans laquelle il se trouve vers la collection correspondant à l'icône sélectionné.

4.5.5.2 LES COMMANDES NE PORTANT PAS SUR UN ICONE

4.5.5.2.1 EDITER

Lorsqu'on sélectionne la commande EDITER, l'éditeur de texte "VI" apparait à l'emplacement de la sous-fenêtre des résumés après quelques secondes (figure 4.9). C'est en fait une autre sous-

fenêtre qui vient se superposer à celle des résumés. Dès lors, il y a possibilité de la déplacer sur l'écran. On peut par exemple la changer d'endroit pour pouvoir avoir un résumé sous les yeux pendant qu'on tape son texte via l'éditeur.

Une fois le texte tapé, on quitte l'éditeur de texte. Le document va alors se mettre automatiquement dans la collection des documents en attente de traitement et l'éditeur disparait de l'écran .

4.5.5.2.2 QUITTER

La commande QUITTER permet de sortir du programme . Une fois cette commande sélectionnée, le programme demande une confirmation sous la forme d'un message qui apparait dans un rectangle au milieu de l'écran pour ne pas altérer celui-ci en cas de non-confirmation. Selon que vous ayez confirmé ou non votre désir de sortir du programme, celui-ci s'arrêtera ou continuera .

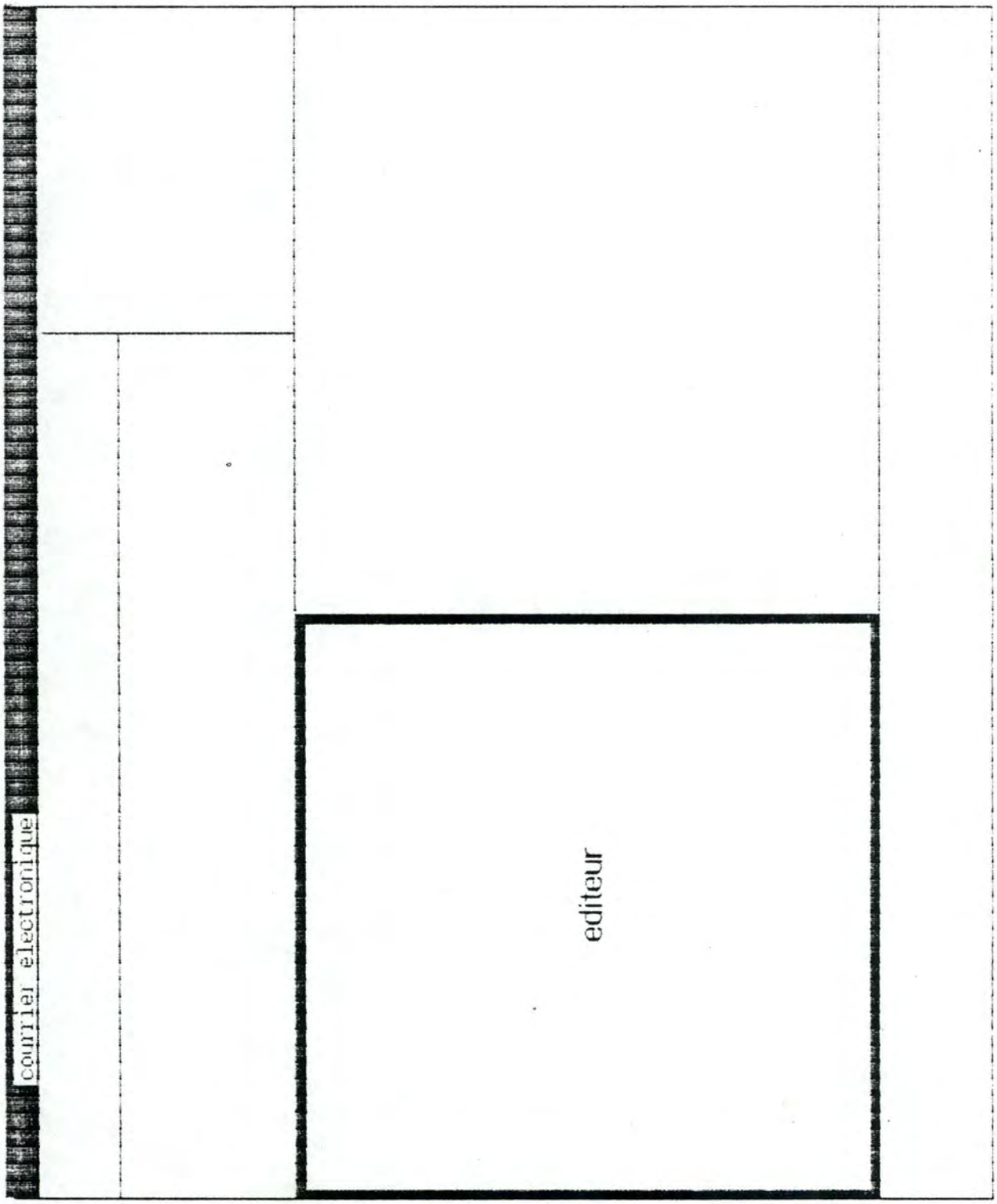


Figure 4.9: SOUS-FENETRE D´EDITION

4.5.6 SOUS-FENETRE DES ICONES

Il existe deux types d'icônes : ceux qui représentent une collection de documents et ceux qui n'en représentent pas.

4.5.6.1 LES ICONES REPRESENTANT UNE COLLECTION DE DOCUMENTS

Ils sont au nombre de sept et portent les noms suivants : boite au lettre, documents en attente, réponses reçues à un document envoyé, père (c-à-d le document reçu qui est à l'origine d'un document envoyé), courrier archivé, poubelle et signataire (figure 4.11). On les utilise pour désigner la collection sur laquelle on veut effectuer une consultation ou dans laquelle on veut transférer un document .

4.5.6.2 LES AUTRES ICONES

Il y en a six et ils se dénomment de la sorte : le bottin (c-à-d la liste des listes des abonnés, et le détail de ces listes), l'imprimante, le mot de passe, l'administration, la boite out et la poste (figure 4.12). Il sont subdivisés en deux sous-types : ceux sur lesquels on ne peut effectuer que des opérations de transfert (imprimante et boite out) et ceux sur lesquels on ne peut effectuer que des opérations de consultation (bottin,mot de passe, administration et poste).

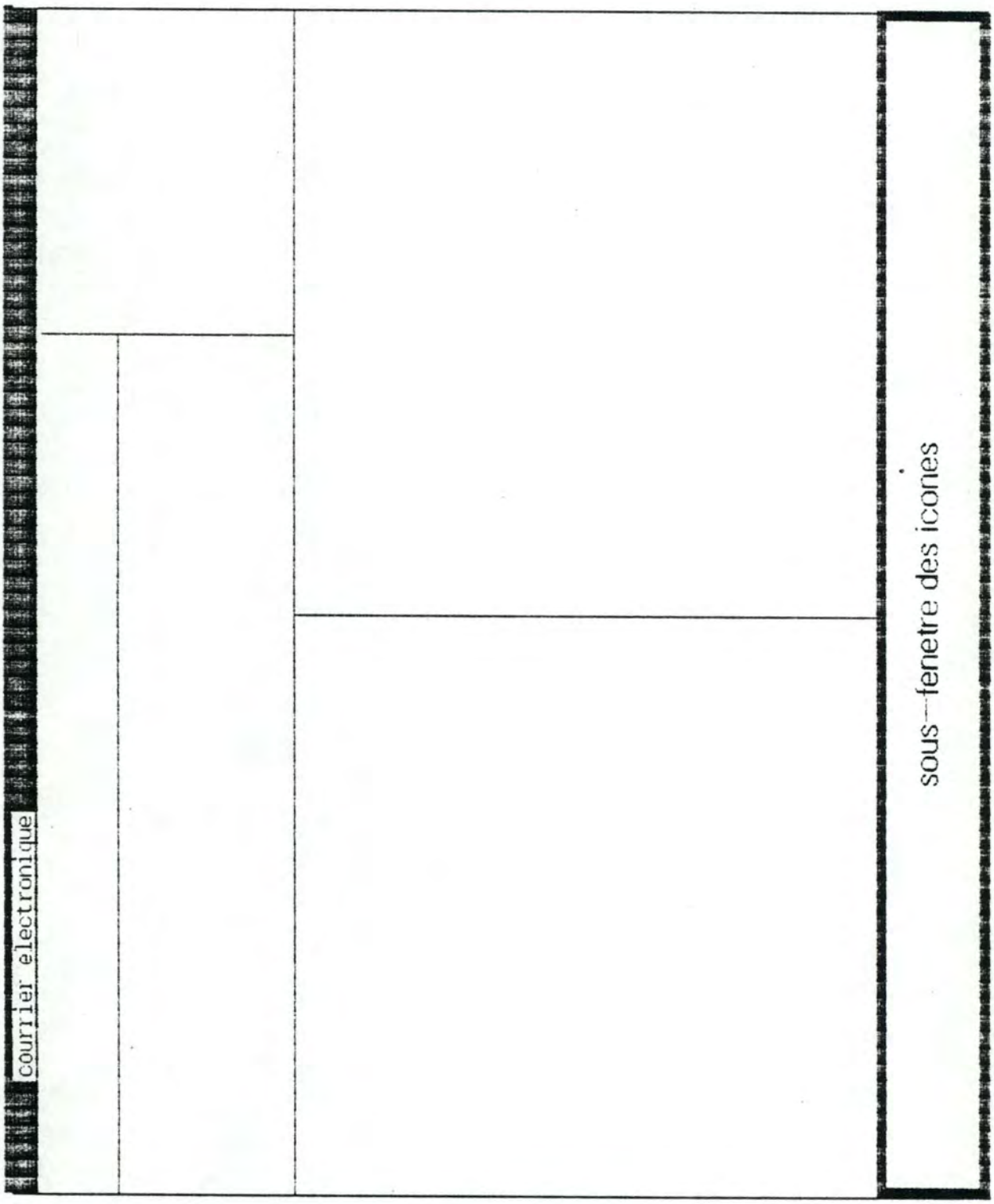
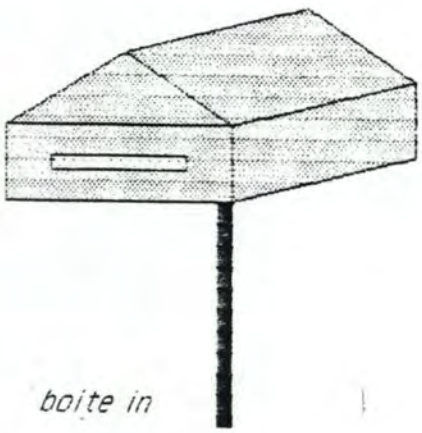
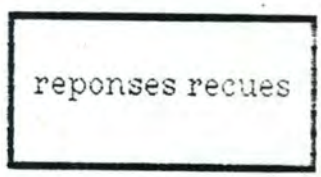


Figure 4.10: SOUS-FENETRE DES ICONES

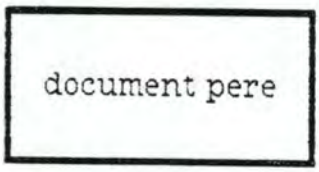


boite in

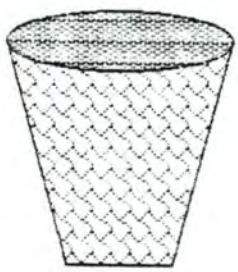
documents en attente



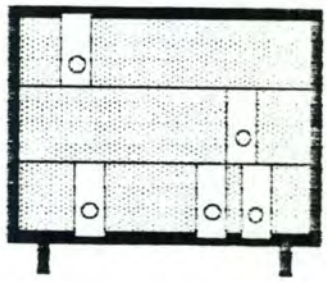
reponses recues



document pere



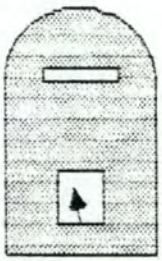
poubelle



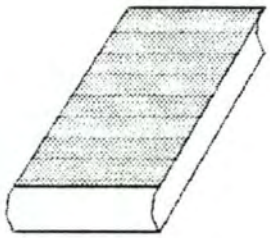
archives

signataire

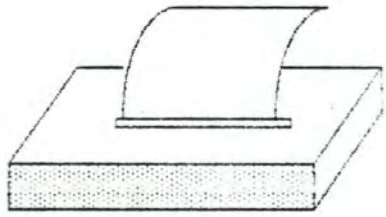
Figure 4.11: ICONES REPRESENTANT UNE COLLECTION DE DOCUMENTS



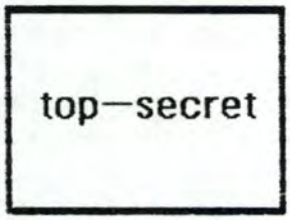
boite out



bottin

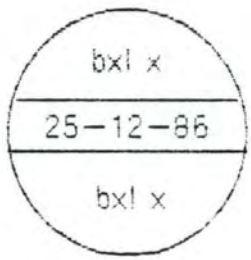


imprimante



mot de passe

administration



poste

Figure 4.12: ICONES NE REPRESENTANT PAS UNE COLLECTION DE DOCUMENTS

4.5.6.3 REGLE GENERALE

Un icône va toujours de pair avec une commande. Il est dès lors obligatoire de sélectionner une commande avant de sélectionner un icône . Si on sélectionne un icône seul, un message d'erreur apparaitra dans la fenêtre des messages .

Les commandes et les icônes forment donc des couples. Cependant, tous les couples ne sont pas permis (cfr ci-dessus).

C'est le cas notamment de l'icône BOITE OUT, sur lequel on ne peut faire qu'un transfert, de même que l'icône imprimante . Par contre, on ne peut pas faire de transfert sur l'icône BOTTIN . Il existe donc des possibilités d'erreur . Quand on sélectionne un mauvais couple, un message d'erreur apparait dans la sous-fenêtre des messages.

4.6 RESPECT DES QUALITES D'UN BON INTERFACE

Pour montrer qu'on respecte bien les qualités et les recommandations énoncées au chapitre 2, nous allons suivre le même plan que celui suivi dans ce chapitre. Nous montrerons d'abord que nous possédions les pré-requis pour concevoir notre interface. Ensuite, nous reprendrons les huit qualités d'un bon interface pour voir si elles sont bien respectées. Nous poursuivrons en parlant du dialogue, mais sans reprendre dans les détails toutes les recommandations. Enfin, nous montrerons que les recommandations supplémentaires pour un logiciel de courrier électronique sont bien suivies.

4.6.1 RESPECT DES PRE-REQUIS

Les deux pré-requis pour la conception d'un bon interface sont de connaitre la tâche et de connaitre les utilisateurs.

Pour ce qui est de la tâche, elle était clairement définie dans le mémoire qui est à la base du nôtre. Notre travail consiste

en l'amélioration de l'interface d'un logiciel existant, et dès lors, la définition exacte de la tâche ne nous incombe pas.

En ce qui concerne les utilisateurs, le but de notre mémoire était de faire un interface pour des utilisateurs inexpérimentés. La classe des utilisateurs était donc elle aussi bien définie.

4.6.2 RESPECT DES QUALITES

Dans la conception de notre interface, nous nous sommes efforcé d'obtenir les qualités requises. Nous allons les reprendre une à une, et dire comment nous y sommes arrivé.

4.6.2.1 COMPATIBILITE

Nous avons repris le même vocabulaire que dans le mémoire de départ. Ce vocabulaire est un vocabulaire couramment employé, et qui sera familier pour l'utilisateur. De plus, créant ses documents lui-même avec l'éditeur de texte, l'utilisateur pourra leur donner le format qu'il désire. Ainsi il ne sera pas dépaysé par rapport à la réalité et les performances suivront.

4.6.2.2 HOMOGENEITE

La qualité d'homogénéité est tout à fait respectée. Les mêmes tâches avec les mêmes conditions sont toujours exécutées par les mêmes séquences d'actions. Le travail de l'utilisateur est encore

d'avantage facilité par le fait qu'il existe des sous-fenêtres avec des fonctions bien spécifiques. Il fait donc toujours les mêmes actions aux mêmes endroits.

4.6.2.3 CONCISION

Les commandes et les objets sur lesquels celles-ci s'appliquent sont présents à l'écran durant toute l'exécution du programme. L'utilisateur ne devra dès lors pas se remémorer les commandes à effectuer. Il en est de même pour les commande-menus qui s'affichent toutes sans exception lorsqu'on les demande.

4.6.2.4 FLEXIBILITE

L'interface que nous avons conçu, l'a été pour des utilisateurs inexpérimentés. Ceci étant le but du mémoire, nous n'avons pas prévu que l'utilisateur puisse passer à un autre type d'interface lorsqu'il saurait parfaitement s'en servir. De toute manière, tout en recherchant à faciliter le travail de l'utilisateur, nous n'en n'avons pas moins essayé d'obtenir un très bon niveau de performance, grâce notamment au travail en parallèle.

4.6.2.5 GUIDAGE

Vu la structure de la sous-fenêtre des messages, l'utilisateur du logiciel sait à tout instant où il se trouve dans

l'exécution de ses tâches. De plus, à chacune de ses actions, correspond un message, soit qui confirme la réalisation de l'action, soit qui l'interdit. Des exemples illustrant cet aspect seront vus plus loin. Enfin, un ensemble exhaustif des commandes et des commande-menus qu'il peut employer lui est constamment disponible, et le guide vers les différentes possibilités qui lui sont offertes.

4.6.2.6 CHARGE INFORMATIONNELLE

Dans l'interface conçu, la sélection d'une commande ou d'une commande-menu se fait toujours par positionnement du curseur sur celle-ci et pression d'un bouton de la souris. On minimise donc le nombre de touches à enfoncer et on limite par ce fait même la probabilité de commettre des erreurs.

4.6.2.7 CONTROLE EXPLICITE

Cette qualité est entièrement respectée : chaque action de l'utilisateur est suivie d'un message quant à la réussite ou non de son action. C'est lui qui provoque les changements de configuration de l'écran.

4.6.2.8 GESTION DES ERREURS

Les qualités ci-dessus ayant été respectées, le nombre de possibilités d'erreurs est donc minimisé. De plus, lorsqu'il

commet une erreur (par exemple lorsqu'il veut exécuter une opération interdite), un message lui indique cette erreur et dès lors il peut la corriger.

4.6.3 RESPECT DES RECOMMANDATIONS POUR LE DIALOGUE

En ce qui concerne le dialogue, les recommandations ont été en grande partie suivies. Les voici reprises, non en détail, mais par type de recommandations :

- le dialogue est à l'initiative de l'ordinateur;
- le dialogue a été conçu par rapport au type d'utilisateurs (c'est-à-dire des utilisateurs inexpérimentés);
- les mêmes dialogues sont utilisés pour les mêmes fonctions;
- nous utilisons un dialogue par menus;
- nous avons limité le nombre et la longueur des entrées de telle sorte que les possibilités d'erreurs soient fortement réduites mais que le programme reste quand même compréhensible, vu l'inexpérience des utilisateurs;
- la division de l'écran en sous-fenêtres ayant chacune leurs fonctions spécifiques donne à celui-ci une grande clarté;
- l'information fournie à l'utilisateur est pertinente : tout ce qui est utile lui est fourni, et tout ce qui lui est fourni lui est utile;
- l'affichage a été conçu de telle manière qu'il n'y ait aucune confusion possible

Nous devons encore faire une remarque en ce qui concerne le

temps de réponse : notre programme étant un interface tournant indépendamment du logiciel de courrier électronique, il est impossible de pouvoir déterminer le temps de réponse.

4.6.4 RESPECT DES RECOMMANDATIONS POUR UN COURRIER ELECTRONIQUE

Nous pouvons simplement dire que les cinq recommandations qui ont été formulées ont été respectées.

4.7 EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

4.7.1 INTRODUCTION

Nous allons analyser l'interface lors du déroulement de la réception d'un message. Pour une compréhension plus facile de l'exemple, celui-ci sera vu à l'aide de figures . Celles-ci seront la représentation des différents écrans apparaissant lors de l'exécution du programme . Pour chaque écran, une explication complémentaire sera donnée .

Contrairement à la réalité, on admettra que le curseur puisse se trouver plusieurs fois sur un même écran : de la sorte, on limite le nombre d'écrans qui est déjà très important. Cependant, pour la compréhension, les différentes positions de la souris sur un même écran seront numérotées dans l'ordre chronologique .

4.7.2 DEROULEMENT DE L'EXECUTION

Les documents reçus par un abonné se trouvent dans la boite-in . Pour pouvoir être au courant des documents qu'on lui a envoyés, l'utilisateur doit donc faire une consultation de la collection des documents se trouvant dans cette boite-in . Il faut dès lors que le curseur soit déplacé à l'aide de la souris et positionné sur CONSULTER (figure 4.13) dans la sous-fenêtre des commandes . Ceci fait, le curseur se positionne automatiquement dans la sous-fenêtre des icônes et il ne reste plus à l'utilisateur qu'à le déplacer sur BOITE-IN (figure 4.13) .

Lorsqu'il a fait cela, deux choses se produisent (figure 4.14) :

- le message "CONSULTATION BOITE-IN" apparait dans la sous-fenêtre des messages pour indiquer à l'utilisateur ce qu'il est en train de faire .
- le premier document de la collection de la boite-in et le résumé de ce document apparaissent respectivement dans la sous-fenêtre des documents et dans celle des résumés .

A ce stade, l'utilisateur peut sélectionner le document ou le résumé qu'il désire traiter, a l'aide du menu (figure 4.14) . Grâce à la souris, il peut demander à voir le PREMIER document, le DERNIER, le PRECEDENT ou le SUIVANT, ou bien il peut sélectionner le document présent . Supposons qu'il choisisse de voir le document suivant en sélectionnant la commande-menu SUIVANT (figure

4.14) : le deuxième document de la collection apparait alors dans la sous-fenêtre des documents, tandis que le résumé correspond toujours au premier document (figure 4.15) .

L'utilisateur peut alors décider de sélectionner ce document au moyen de la commande-menu SELECTION (figure 4.15) . Cela a pour effet de faire apparaitre dans la sous-fenêtre des messages "DOCUMENT SELECTIONNE" (figure 4.16), le reste de l'écran restant inchangé .

Ayant sélectionné un document reçu, on peut alors le transférer dans une autre collection. Si, par exemple, l'utilisateur veut le mettre dans la poubelle, il positionne successivement le curseur sur TRANSFERT dans la sous-fenêtre des commandes, et sur POUBELLE dans celle des icônes (figure 4.16) . Suite à cela, apparait, dans la partie gauche de la sous-fenêtre des messages, "TRANSFERT EFFECTUE", et dans la partie droite, "TRANSFERT POUBELLE" (figure 4.17) . Si l'utilisateur avait voulu mettre le document dans une collection vers laquelle un transfert était interdit, le message "TRANSFERT INTERDIT" aurait indiqué que c'était une action qu'on ne peut pas exécuter .

4.8 CONCLUSION

Le fait d'utiliser le SUN et ses particularités techniques (écran, souris ...) nous a permis de développer un interface plus évolué que celui qui existait dans le mémoire de départ. Etant

donné qu'il devait être conçu pour des utilisateurs inexpérimentés, nous avons opté pour la facilité d'utilisation plutôt que pour la sophistication de l'interface. En effet, nous avons essayé de limiter au maximum les manipulations à faire pour faciliter le travail et pour prévenir les nombreuses possibilités d'erreurs inhérentes à cette classe d'utilisateurs. Nous avons enfin essayé de suivre au maximum les recommandations pour obtenir un bon interface. Reste à voir, maintenant, comment après l'avoir ainsi conçu, nous l'avons implémenté.

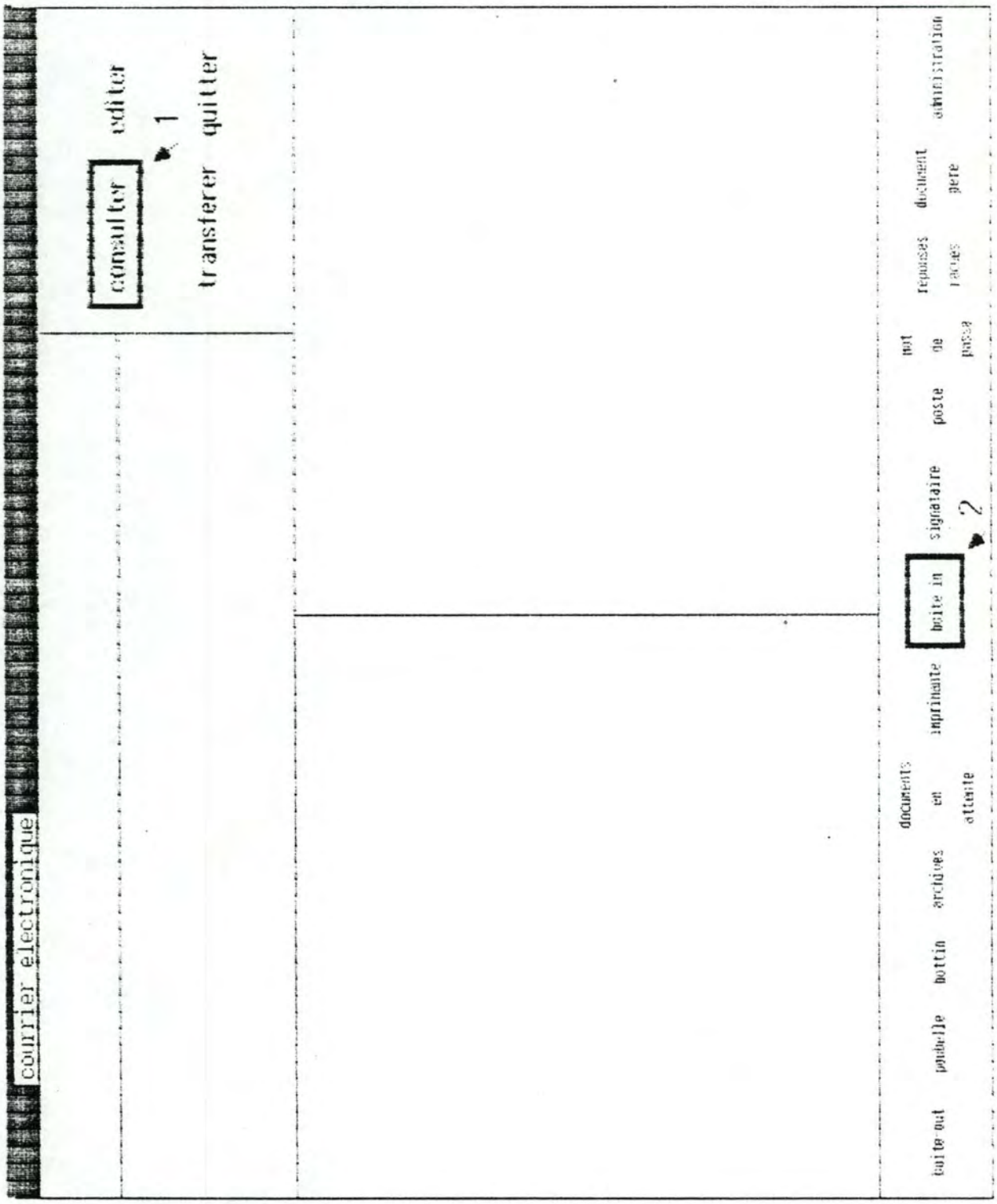


Figure 4.13: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

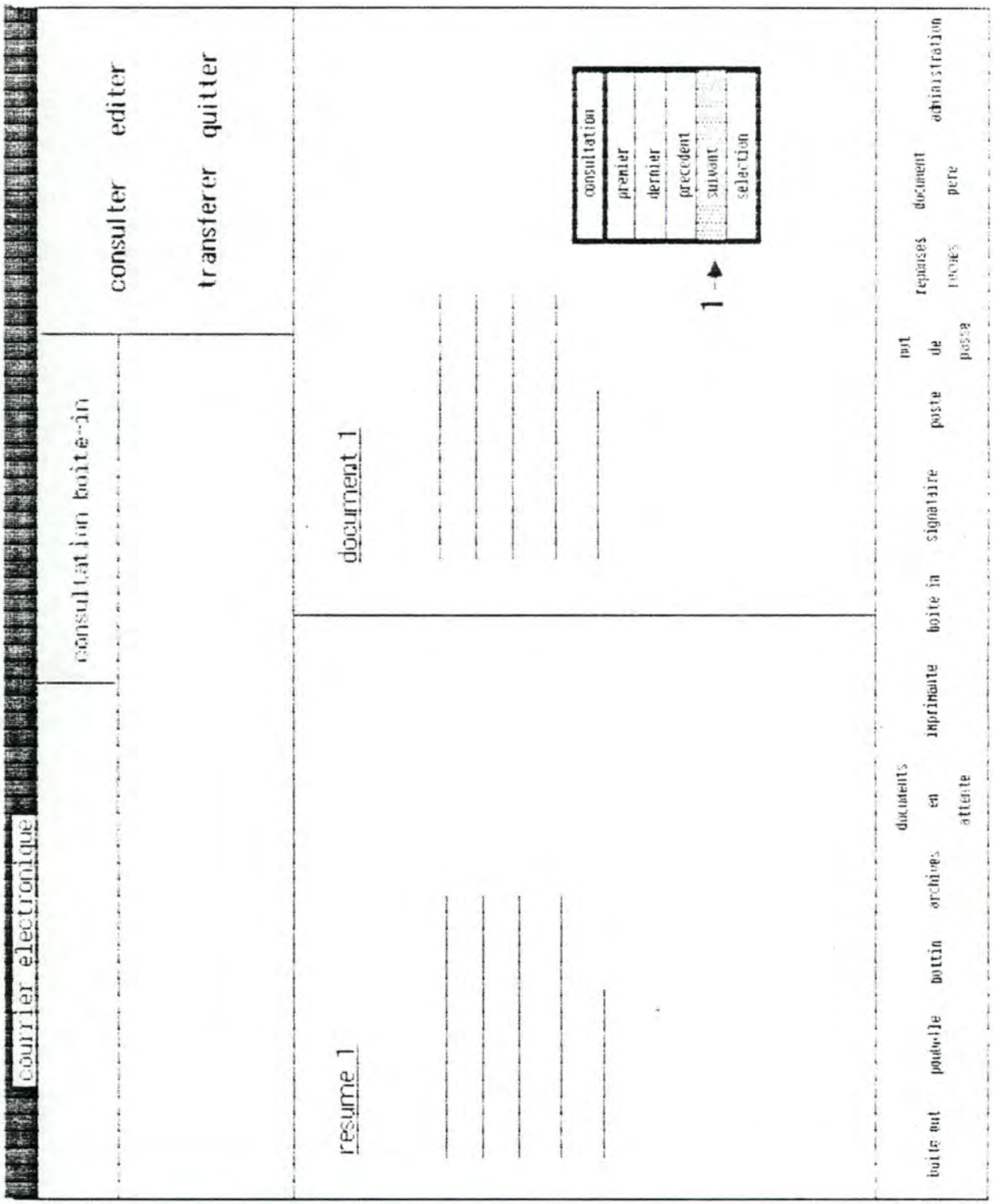


Figure 4.14: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

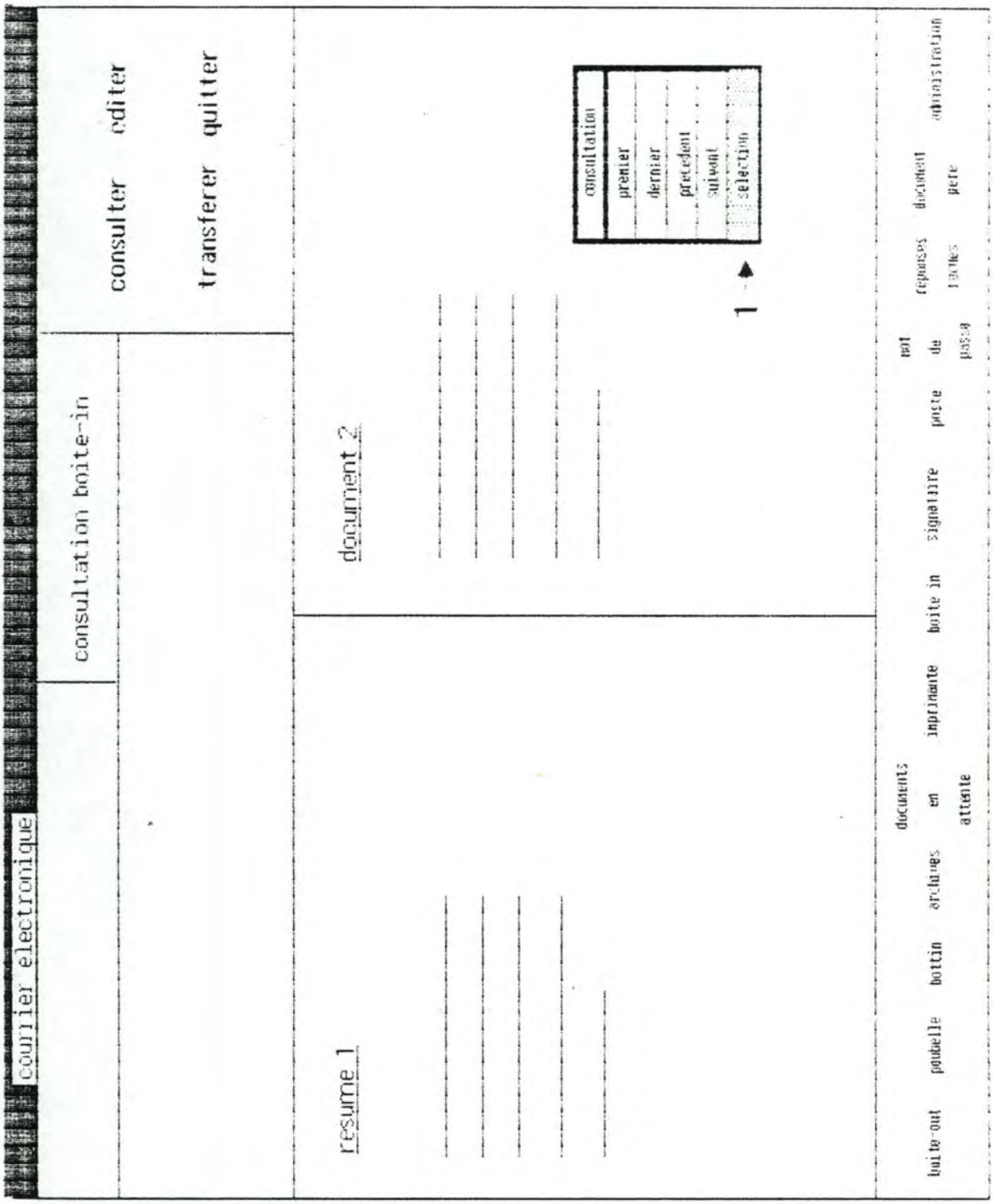


Figure 4.15: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

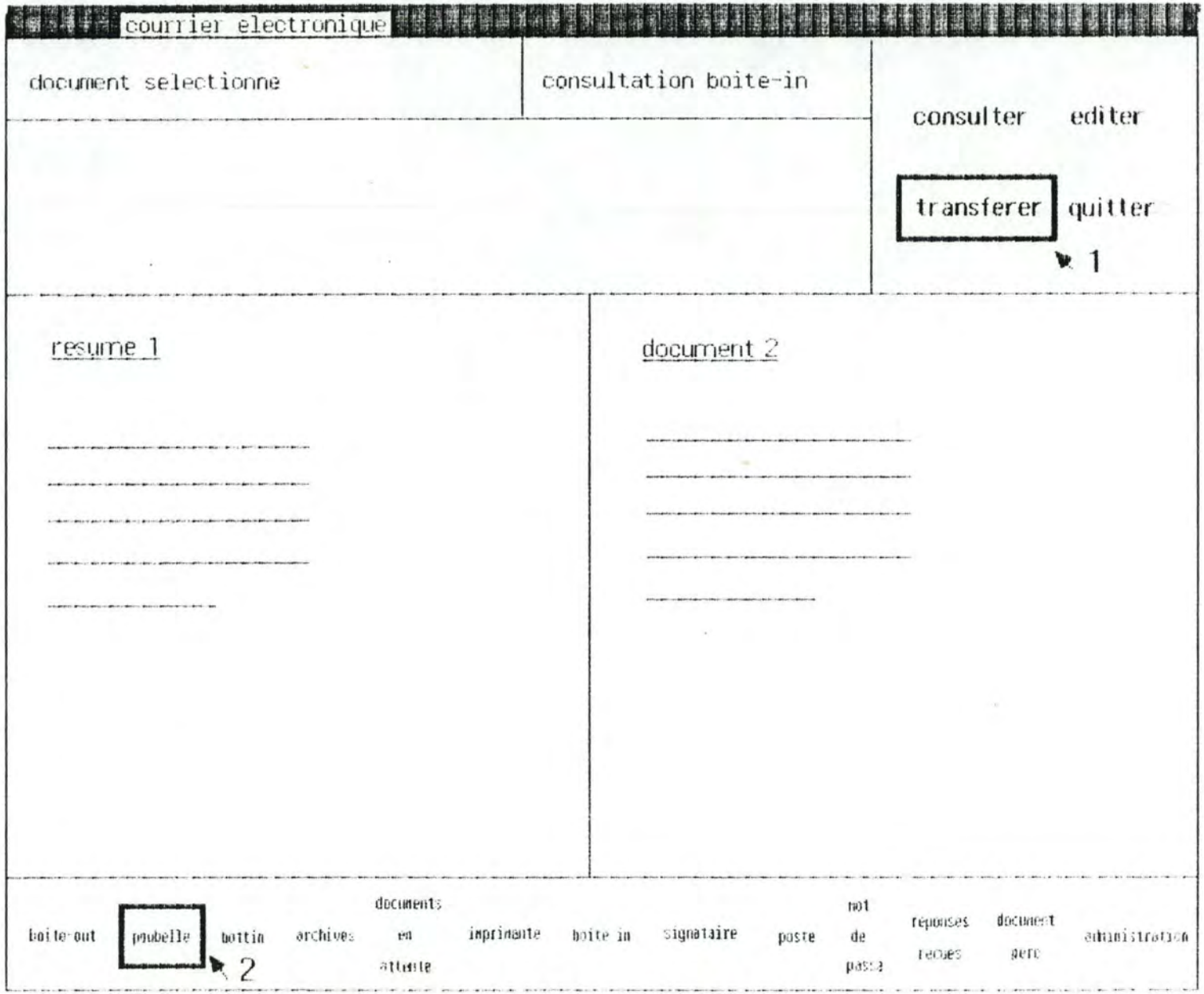


Figure 4.16: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

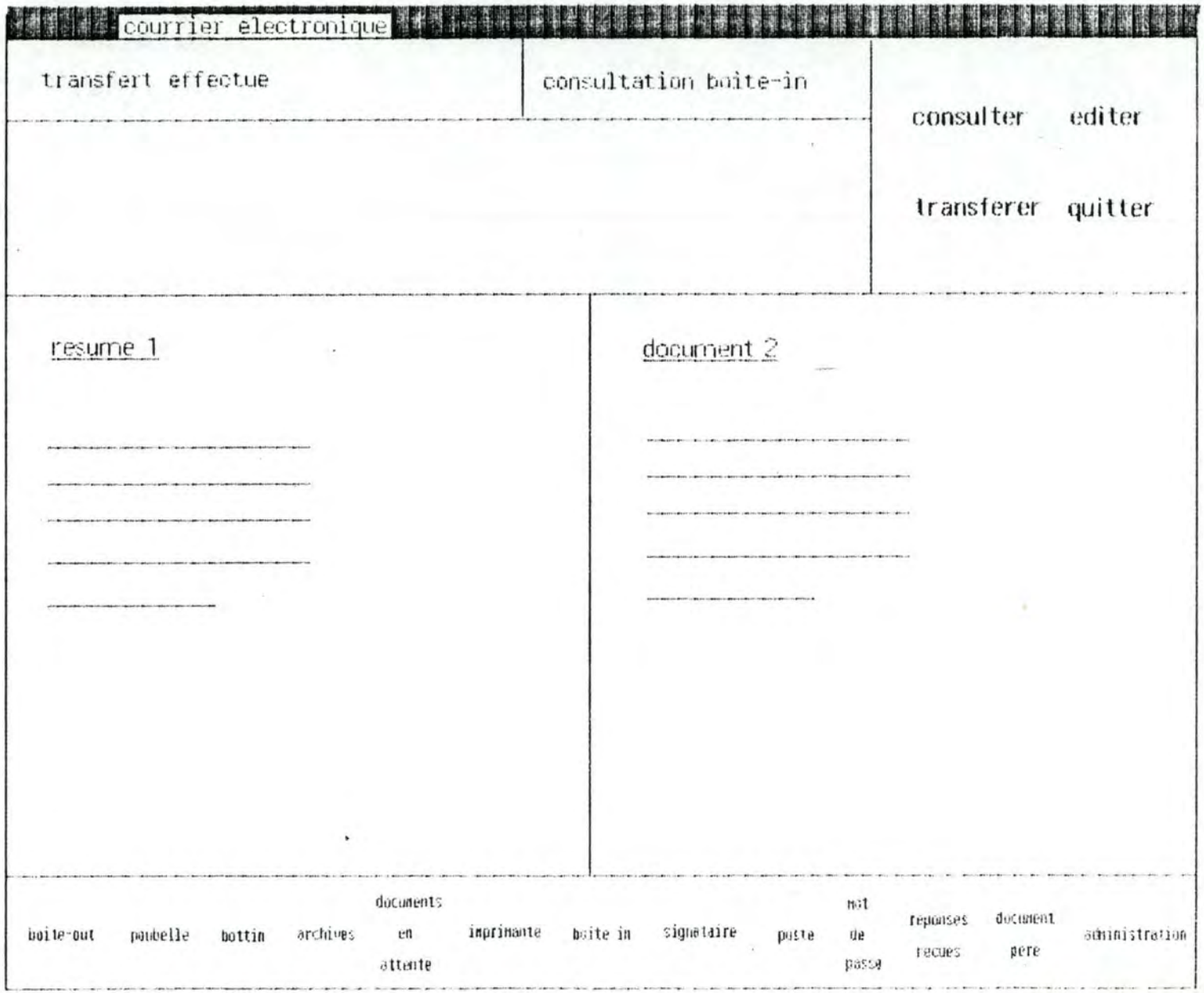


Figure 4.17: EXEMPLE DE FONCTIONNEMENT DE LA RECEPTION D'UN MESSAGE

5 ANALYSE ORGANIQUE

5.1 INTRODUCTION

Le but du mémoire est d'améliorer l'interface d'un programme de courrier électronique qui a déjà été écrit. C'est pourquoi notre programme ne comporte que les fonctions nécessaires à l'interface, plus quelques fonctions nécessaires pour simuler celles du programme de courrier électronique, afin de pouvoir se rendre compte du fonctionnement général du programme.

Pour avoir un programme complet, il serait nécessaire de transférer le programme de départ sur une machine SUN, puis d'intégrer notre programme dans le programme de départ. Ceci poserait quelques problèmes, notamment à cause de la différence de machines (IBM PC pour le mémoire de départ, SUN pour le nôtre), entre les langages (le premier programme est écrit en PASCAL, le second en C), et entre les environnements dans lesquels ils ont été réalisés (systèmes d'exploitation différents, ...).

Dans ce chapitre, nous allons d'abord donner les grands principes d'implémentation que nous avons adoptés, puis nous spécifierons les fonctions principales du programme : vu la longueur du programme - 6000 lignes - et les similitudes entre de nombreuses fonctions, il ne nous est pas paru nécessaire de spécifier l'ensemble des fonctions du programme.

5.2 PRINCIPES GENERAUX DE L'IMPLEMENTATION

Comme vu au chapitre 4 au point Techniques de programmation des tools, les programmes qui utilisent le principe des fenêtres ont une structure identique, car ils sont tous "event-driven" (conduits par les événements). C'est donc le cas pour notre programme également. Voyons maintenant en détail en quoi consiste cette structure.

Tout d'abord, il est obligatoire de définir la fenêtre, et les sous-fenêtres qui la composent. Ceci se fait au moyen de fonctions prédéfinies au niveau "Suntool". Ces fonctions ont de nombreux paramètres pour définir la taille, la position ... de la fenêtre ou des sous-fenêtres. Il existe deux types de fonctions pour la définition d'une sous-fenêtre, celles qui définissent des sous-fenêtres prédéfinies par le système, et celle qui définit une sous-fenêtre tout à fait propre à l'application. Dans notre cas, il existe six sous-fenêtres : deux sont des sous-fenêtres à panneaux de commande, et les quatre autres sont des sous-fenêtres qui sont propres à ce programme (il n'existait pas de sous-fenêtres prédéfinies par le système qui auraient pu convenir pour faire ce qui devait être fait par ces sous-fenêtres).

Une fois la fenêtre et les sous-fenêtres définies, il faut initialiser les sous-fenêtres. Cette initialisation peut être très différente selon le type de la sous-fenêtre. Pour les deux sous-fenêtres à panneaux de commande, l'initialisation consiste à

définir le type et l'emplacement dans la sous-fenêtre des différents panneaux dont on a besoin dans le programme (par exemple pour la sous-fenêtre des commandes, on a défini quatre panneaux qui sont constitués chacun d'un seul label - CONSULTER, TRANSFERER, EDITER, QUITTER - et on a défini leur position dans la sous-fenêtre de manière à ce qu'ils soient centrés), et également pour chaque panneau la fonction à exécuter quand il a été sélectionné par l'utilisateur. Pour les sous-fenêtres propres au programme, il faut définir les fonctions à activer quand cette sous-fenêtre est manipulée (changement de place, de taille, recouvrement) ou en cas d'entrée dans cette sous-fenêtre. Il faut également définir les entrées valides pour cette sous-fenêtre en définissant un masque des entrées. Il est à noter que ce qui est fait lors de l'initialisation des sous-fenêtres propres au programme est fait également lors de l'initialisation des sous-fenêtres prédéfinies, mais automatiquement (par le système).

Ensuite, on installe le "tool" (c-à-d on affiche à l'écran le "tool" dans son état initial et on ajoute ce "tool" à la base de donnée des "tools") par une fonction du niveau "suntool", "tool_install", et on entre dans la boucle principale du programme, la fonction "tool_select". A partir de ce moment, le programme attend une entrée quelconque, identifie la sous-fenêtre dans laquelle le curseur se trouve au moment de l'entrée, exécute la fonction prévue dans le programme pour ce type d'entrées dans cette sous-fenêtre, puis recommence la boucle en attendant l'entrée suivante. Il reste maintenant à définir les différents événements que l'utilisateur peut provoquer, et les traitements à

effectuer pour répondre à ces entrées.

Pour chaque sous-fenêtre, on a donc une fonction qui est appelée par "tool-select" quand une entrée se produit dans cette sous-fenêtre. Cette fonction lit l'entrée, puis, en fonction de cette entrée, effectue les opérations nécessaires pour y répondre.

Nous allons illustrer ce principe par l'exemple de la lecture des paramètres dans la sous-fenêtre des paramètres. Les fonctions qui s'occupent de la lecture de ces paramètres prennent grosso modo la moitié du programme, soit 3000 lignes. Dans un programme classique, on peut estimer à 500 le nombre de lignes qui auraient été nécessaires pour remplir le même travail. Ceci souligne la différence de programmation, due au fait que le programme est "event-driven", et qu'il est possible de travailler en parallèle dans les différentes sous-fenêtres.

La fonction qui s'occupe de la lecture des paramètres est celle qui s'occupe des entrées dans la sous-fenêtre des paramètres, car les seules entrées valides dans cette sous-fenêtre sont les caractères des paramètres que l'utilisateur doit entrer. Cette fonction commence donc par lire l'entrée. Le point important à souligner à ce niveau est qu'une entrée dans la sous-fenêtre des paramètres correspond à UN caractère. La fonction est donc appelée chaque fois que l'utilisateur tape UN caractère lorsque le curseur est dans cette sous-fenêtre. Ceci rend le travail de lecture des paramètres très différent de la lecture classique d'une variable alphanumérique. En effet, classiquement,

on lit une variable en utilisant une fonction de lecture qui lit les caractères un après l'autre jusqu'à ce que l'utilisateur tape un caractère de fin de variable, souvent RETURN. Mais le programme reste dans cette fonction tant que l'utilisateur n'a pas tapé ce caractère de fin, et quand le programme sort de cette fonction, cela veut dire que la variable est entièrement tapée. Ceci n'est plus le cas ici : en effet, on appelle la fonction de lecture pour CHAQUE caractère de la variable alphanumérique que l'utilisateur rentre au clavier. Ceci parce que entre chaque caractère tapé, l'utilisateur peut faire autre chose dans une autre sous-fenêtre (grâce au parallélisme) et on peut donc être amené à exécuter d'autres fonctions entre la lecture de deux caractères.

Comment se présente maintenant la fonction de lecture ? Prenons l'hypothèse qu'une seule variable doit être lue (en fait, dans la majorité des cas, plusieurs variables doivent être lues une après l'autre, ce qui complique le problème), par exemple un nom d'abonné. La fonction de lecture des paramètres reçoit donc un premier caractère. A ce moment, deux possibilités : soit ce caractère est le premier de la réponse de l'utilisateur à une demande de paramètre (l'utilisateur est à un point du programme où il doit donner un nom d'abonné, et le libellé "nom de l'abonné : _" est affiché dans la sous-fenêtre des paramètres), soit ce caractère est tapé par l'utilisateur alors qu'on ne lui demandait rien, et il n'a donc aucune valeur. Pour savoir dans lequel de ces deux cas on se trouve, on utilise des indicateurs (un par paramètre) qui sont initialisés au début du programme à 0. Quand

on doit demander un paramètre à l'utilisateur, on positionne l'indicateur correspondant à ce paramètre à 1, et quand l'utilisateur a fini de rentrer ce paramètre, on repositionne son indicateur à 0. Dans notre exemple, quand on est arrivé dans le programme à un moment où il faut demander comme paramètre le nom d'abonné, on a positionné un indicateur, nom_abonné, à 1. Quand un indicateur est à 1, il est interdit d'en mettre un autre à 1, de telle manière qu'il n'y ait à un moment donné jamais qu'un seul indicateur positionné à 1 (on ne demande de toute façon jamais qu'un seul paramètre à la fois puisqu'il n'y a qu'une seule sous-fenêtre des paramètres). De cette manière, on peut savoir quel paramètre l'utilisateur est en train de donner en testant les indicateurs : si ils sont tous à 0, c'est que l'utilisateur entre des caractères qui ne correspondent à rien, et on peut les ignorer, sinon un seul indicateur est à 1 et on sait déterminer alors quel paramètre l'utilisateur donne au moment où la fonction de lecture des paramètres reçoit le premier caractère de la variable.

On sait donc maintenant à quel paramètre ce caractère correspond car on a positionné l'indicateur nom_abonné à 1. Mais il ne faut pas oublier que la fonction de lecture est appelée pour CHAQUE caractère. Il est donc nécessaire de conserver le caractère déjà tapé dans une variable qui ne sera pas modifiée par le fait que l'on appelle la fonction de lecture pour chaque caractère. Ce type de variables s'appelle "variable globale". Elles sont définies en dehors du programme, leur valeur est maintenue tout au long du programme, quelle que soit la fonction

dans laquelle on se trouve, et cette valeur peut être modifiée dans n'importe quelle fonction, sans pour cela la mettre comme paramètre de cette fonction. Ce n'est pas le cas des variables "locales", qui ne gardent leur valeur que durant l'exécution de la fonction dans laquelle elles sont définies, et que nous ne pouvions pas utiliser ici puisque la fonction de lecture est appellée pour chaque caractère.

Il est donc nécessaire de conserver les caractères tapés dans une variable globale, et chaque fois qu'un caractère est tapé, on le rajoute aux prédédents jusqu'à ce que l'utilisateur tape le caractère de fin de variable (RETURN). A ce moment, la variable est entièrement tapée, on peut donc repositionner l'indicateur correspondant à cette variable à 0 (dans l'exemple, on remet nom_abonné à 0).

Pour une question évidente de clarté et de facilité pour l'utilisateur, il est indispensable que, quand celui-ci tape un caractère, il s'imprime sur l'écran à l'endroit voulu, c-à-d à côté du caractère précédent ou à côté du libellé du paramètre s'il s'agit du premier caractère donné par l'utilisateur pour ce paramètre (de la même manière que lorsque l'on tape sur un terminal classique). De plus il faut lui accorder le droit à l'erreur, et donc lui permettre d'effacer un ou plusieurs caractères pour corriger ce qu'il a tapé. Ceci amène quelques difficultés supplémentaires, car il faut calculer où un caractère doit être imprimé, c-à-d les coordonnées dans la sous-fenêtre. Il est donc nécessaire de calculer ces coordonnées pour chaque

caractère, et pour cela il faut connaitre les coordonnées du caractère précédent qui a été imprimé. D'où la nécéssité de conserver les coordonnées en variable globale.

5.3 DICTIONNAIRE DES VARIABLES

TEXT : chaine de caractères à imprimer dans la sous-fenêtre des messages

PART : entier indiquant dans quelle partie de la sous-fenêtre TEXT doit être imprimé (1 = partie gauche, 2 = partie droite)

les témoins d'édition : EDIT_EN_COURS, EDIT_NOTE, EDIT_MOD, EDIT_CORPS,EDIT_REP, EDIT_RAPPEL : ces témoins indiquent si une édition est en cours : lorsqu'un témoin est à 1, l'édition est en cours

NBCAR : compteur de caractères utilisé lors de la lecture des paramètres dans la sous-fenêtre des paramètres

NBMOT : numéro du paramètre qui est demandé dans la sous-fenêtre des paramètres

ATT_REP : témoin de la saisie des paramètres d'envoi d'un rappel dans la sous-fenêtre des paramètres

FEN : sous-fenêtre des paramètres

FEN2 : sous-fenêtre des documents

FEN3 : sous-fenêtre des résumés

FEN_MES : sous-fenêtre des messages

OP1 : sous-fenêtre des commandes

OP2 : sous-fenêtre des icônes

FEN_PIXWIN : structure définissant l'ensemble des points de FEN

FEN2_PIXWIN : structure définissant l'ensemble des points de FEN2

FEN3_PIXWIN : structure définissant l'ensemble des points de FEN3

FEN_MES_PIXWIN : structure définissant l'ensemble des points de FEN_MES

INFO_SELECT : tableau de caractères contenant l'information que l'utilisateur a sélectionné avec la souris dans la sous-fenêtre des documents, lors de la consultation du bottin (c-à-d un nom d'abonné, ou un nom de liste d'abonnés)

les tableaux de listes d'abonnés : TABABBE, TABPROF, TABELEV, TABALL : tableaux de chaines de caractères contenant les noms des abonnés

les longueurs des tableaux de listes d'abonnés : LG_TABABBE, LG_TABPROF, LG_TABELEV, LG_TABALL

CONT_FEN2 : tableau contenant des structures composées d'une chaine de caractères et des coordonnées d'un rectangle. Ce tableau est utilisé lors de la consultation du bottin, il contient les chaines de caractères qui sont affichées dans la sous-fenêtre des documents (qui sont des noms d'abonnés ou des noms de liste d'abonnés), et leur position sur l'écran, de telle manière que,

quand l'utilisateur appuye sur le bouton gauche de la souris, on puisse calculer sur quel nom d'abonné le curseur se trouvait (car on connait les coordonnées du curseur à tout moment)

LG_CONT_FEN2 : longueur du tableau CONT_FEN2

PASS_ICONE : numéro du dernier icône que l'utilisateur a choisi après avoir choisi la commande consultation

IEVENT_FEN : structure contenant le dernier événement (ou entrée) qui s'est produit dans FEN (un événement peut être un caractère tapé par l'utilisateur, un bouton de la souris, ou un timer)

IEVENT_FEN2 : structure contenant le dernier événement (ou entrée) qui s'est produit dans FEN2 (un événement peut être un caractère tapé par l'utilisateur, un bouton de la souris, ou un timer)

IEVENT_FEN3 : structure contenant le dernier événement (ou entrée) qui s'est produit dans FEN3 (un événement peut être un caractère tapé par l'utilisateur, un bouton de la souris, ou un timer)

IEVENT_FEN_MES : structure contenant le dernier événement (ou entrée) qui s'est produit dans FEN_MES (un événement peut être un caractère tapé par l'utilisateur, un bouton de la souris, ou un timer)

les structures définissant les menus : un menu est défini par une structure qui reprend le nom de ce menu, l'ensemble des commandes qui interviennent dans ce menu, le nombre de ces commandes, et un pointeur vers le menu suivant en cas de menus en cascade

OKFEN : témoin indiquant si la sous-fenêtre des paramètres est

disponible (si OKFEN est égal à 1, la sous-fenêtre est disponible, sinon OKFEN est égal à 0)

PERE_PRES_RES : témoin indiquant la présence ou non d'un document père dans la sous-fenêtre des résumés

PERE_PRES_DOC : témoin indiquant la présence ou non d'un document père dans la sous-fenêtre des documents

ARCH_UNIQ : témoin indiquant si l'on consulte un seul document des archives ou plusieurs

NUMDOC : numéro du document qui est actuellement présent dans la sous-fenêtre des documents

NUMRES : numéro du résumé qui est actuellement présent dans la sous-fenêtre des résumés

DOC : fichier contenant les documents (organisation en séquentiel)

RES : fichier contenant les résumés (organisation en séquentiel)

CORPS : fichier contenant le corps du document que l'on veut modifier

les témoins d'utilisation de la sous-fenêtre des paramètres : ADM, MP, SAISIE_PARAM, ATT_REP, ARCHIVE : ces témoins indiquent quels sont les paramètres qui sont demandés dans la sous-fenêtre des paramètres. Un seul de ces témoins est à 1 à un moment donné, ou alors ils sont tous à 0 (par exemple, si ARCHIVE est à 1, on sait que l'on est en train de demander les paramètres d'un archivage)

SUP : variable temporaire dans laquelle sont stockés les

caractères tapés dans la sous-fenêtre des paramètres, jusqu'à ce que l'utilisateur tape le caractère de fin de variable (RETURN)

NB_MOT_CLES : compteur du nombre de mots-clé déjà donnés par l'utilisateur (il peut en donner 5 au maximum)

ICONx_LABEL : structure représentant l'icône x (x allant de 1 à 13)

PASS_COMMANDE : numéro de la dernière commande choisie par l'utilisateur (1 = CONSULTER, 2 = TRANSFERER, 3 = EDITER)

PRESLIST : témoin indiquant la présence dans la sous-fenêtre des documents d'une liste d'abonné (ce fait résultant d'une consultation du bottin)

DOC_SELECT : numéro du dernier document sélectionné par l'utilisateur

5.4 SPECIFICATION DES FONCTIONS PRINCIPALES

5.4.1 IMPMES

entrées : TEXT
PART

pré-conditions : PART est égal à 1 ou à 2

sorties : aucune

post-conditions : aucune

objectif : imprime dans la partie PART de la sous-fenêtre des messages le message TEXT

fonctions similaires : aucune

5.4.2 SIGC

entrées : les témoins d'édition

pré-conditions : un et un seul des témoins d'édition est à 1

sorties : NBCAR

NBMOT

ATT_REP

les témoins d'édition

post-conditions : tous les témoins d'édition sont à 0 et si EDIT_RAPPEL était égal à 1, ATT_REP = 1, NBCAR = 0, NBMOT = 1

objectif : gère le signal de fin de processus, c-à-d le fait qu'une édition vient d'être terminée, en mettant à 0 les témoins d'édition et en préparant la fenêtre des paramètres si l'édition qui vient de se terminer était l'édition d'un rappel

fonctions similaires : aucune

5.4.3 INIT_FEN

entrées : FEN

FEN_SIGHANDLER

FEN_INPUT

pré-conditions : aucune

sorties : FEN_PIXWIN

FEN

post-conditions : aucune

objectif : crèe pour FEN un tableau de points en mémoire, FEN_PIXWIN, indique que la fonction de gestion des entrées dans FEN s'appelle FEN_INPUT, que la fonction de gestion des signaux de changement de taille et de position s'appelle FEN_SIGHANDLER, et remplit le masque des entrées dans FEN avec les trois boutons de la souris et les caractères du code ASCII

fonctions similaires : INIT_FEN_MES, INIT_FEN2, INIT_FEN3

5.4.4 FEN_SIGHANDLER

entrées : FEN

FEN_PIXWIN

pré-conditions : aucune

sorties : FEN

FEN_PIXWIN

post-conditions : aucune

objectif : gère les signaux de changement de taille et de position de FEN en reprenant l'image en mémoire de FEN (FEN_PIXWIN) et en

la redessinant à l'écran

fonctions similaires : FEN_MES_SIGHANDLER, FEN2_SIGHANDLER, FEN3_SIGHANDLER

5.4.5 IMPRIM_LISTE

entrées : INFO_SELECT
FEN2_PIXWIN
les tableaux de listes d'abonnés
les longueurs des tableaux de listes d'abonnés

pré-conditions : INFO_SELECT est égal à un des noms de listes ou est vide

sorties : INFO_SELECT
FEN2_PIXWIN
CONT_FEN2
LG_CONT_FEN2

post-conditions : INFO_SELECT est égal à une chaine de caractère vide, et FEN2_PIXWIN est modifié selon le contenu d'INFO_SELECT, et CONT_FEN2 est rempli avec les noms des éléments de la liste qui a pour nom le contenu d'INFO_SELECT et leur position dans FEN2_PIXWIN, et LG_CONT_FEN2 prend pour valeur la longueur de cette liste

objectif : imprime dans FEN2 (en modifiant FEN2_PIXWIN) la liste des abonnés dont le nom est égal à la chaine de caractères contenue dans INFO_SELECT, et remplit CONT_FEN2 avec les noms de

cette liste et leur position dans FEN2_PIXWIN, et donne à LG_CONT_FEN2 la valeur de la longueur de cette liste

fonctions similaires : aucune

5.4.6 GES_MENU_CONS_RES

entrées : PASS_ICONE
IEVENT_FEN3
FEN3
les structures définissant les menus
les témoins d'édition
OKFEN
PERE_PRES_RES
ARCH_UNIQ
NUMDOC
NUMRES
CORPS
RES
DOC

pré-conditions : 1 témoin d'édition au maximum est à 1, et CORPS, RES et DOC sont fermés

sorties : NUMRES
NUMDOC
FEN_PIXWIN
FEN2_PIXWIN
FEN3_PIXWIN

OKFEN

PERE_PRES_RES

les témoins d'édition

post-conditions : 1 témoin d'édition au maximum est à 1

objectif : cette fonction est appelée quand le bouton droit de la souris est enfoncé alors que le curseur est dans FEN3. Cette fonction affiche le menu qui correspond à la consultation qui est en cours (celle dont le numéro est PASS_ICONE), puis répond à la commande qui a été choisie par l'utilisateur dans ce menu, si c'est possible (c-à-d après avoir testé si cette commande pouvait être réalisée au moment où elle est demandée).

fonctions similaires : GES_MENU_CONS_DOC, GES_MENU1_FEN2, GES_MENU_ATTENTE, GES_MENU_ADM

5.4.7 IMPRIM_CHAR_SUIV_FEN

entrées : X

Y

CARACT

NB

MP

NBMOT

ADM

pré-conditions : la coordonnée (X,Y) est inclue dans FEN, et CARACT est différent de 13 (c-à-d le code ASCII de RETURN)

sorties : FEN_PIXWIN

DELETION

post-conditions : si CARACT est égal à 127 (le caractère tapé est le caractère d'effacement), DELETION est égal à 1, sinon DELETION est égal à 0

objectif : imprime dans FEN le caractère CARACT à la position (X,Y), et renvoie 1 dans DELETION si le caractère tapé est le caractère d'effacement. Si CARACT représente le caractère d'effacement, efface le dernier caractère tapé si NB est différent de 1. Si NB est plus grand que 30, imprime un message d'erreur (pas plus de 30 caractères). Si l'utilisateur tape un mot de passe ou un code (c-à-d si MP est égal à 1, ou si ADM est égal à 1 et que NBMOT est égal à 1,9,10,11,12 ou 13), imprime le caractère ~.

fonctions similaires : aucune

5.4.8 FEN_INPUT

entrées : FEN

IEVENT_FEN

les témoins d'utilisation de la sous-fenêtre des paramètres

pré-conditions : au maximum 1 des témoins d'utilisation de la sous-fenêtre des paramètres est à 1.

sorties : FEN

les témoins d'utilisation de la sous-fenêtre des paramètres

post-conditions : au maximum 1 des témoins d'utilisation de la sous-fenêtre des paramètres est à 1.

objectif : cette fonction est appelée de manière asynchrone chaque fois qu'un caractère est tapé ou qu'un bouton de la souris est enfoncé, lorsque le curseur est dans FEN (la sous-fenêtre des paramètres). S'il s'agit d'un des boutons de la souris, la fonction ne fait rien. Si l'utilisateur a tapé la touche ESC, on annule la saisie de paramètres en cours en effaçant FEN, en réinitialisant les témoins d'utilisation à 0, et en affichant un message (opération avortée) dans la sous-fenêtre des messages. S'il s'agit d'un caractère autre que ESC, il est traité en fonction du témoin d'utilisation de la sous-fenêtre des paramètres qui est à 1. Ces témoins permettent de déterminer quels sont les paramètres que l'on est en train de demander à l'utilisateur. Pour le traitement du caractère proprement dit, voir la fonction suivante.

fonctions similaires : FEN2_INPUT, FEN3_INPUT, FEN_MES_INPUT

5.4.9 TRAIT_ARCHIVE

entrées : IEVENT_FEN

NBCAR

NBMOT

SUP

NB_MOT_CLES

pré-conditions : IEVENT_FEN est un caractère

sorties : NBCAR

NBMOT

SUP

FEN_PIXWIN

OKFEN

ARCHIVE

NB_MOT_CLES

post-conditions : aucune

objectif : gère la lecture des paramètres qui peuvent être demandés lors de la consultation des documents archivés. Cette fonction est appelée lorsque la fonction FEN_INPUT reçoit un caractère et que le témoin d'utilisation de la sous-fenêtre des paramètres qui est à 1 est ARCHIVE. Cela veut dire que le caractère qui a été tapé est un caractère qui fait partie d'un des paramètres que le programme peut demander lors de la consultation des documents archivés. La fonction détermine d'abord si le caractère est le caractère de fin de variable (RETURN) ou pas. Si oui, on gère ce paramètre selon sa valeur (bien souvent la gestion consiste à déterminer quel sera le paramètre à demander par la suite), puis on prépare la réception du paramètre suivant en donnant le numéro de ce paramètre à NBMOT et en remettant NBCAR à 0 (s'il s'agissait du dernier paramètre à rentrer, on libère la sous-fenêtre des paramètres en mettant OKFEN à 1, et on signale que la saisie des paramètres pour la consultation des documents

archivés est terminée en remettant ARCHIVE à 0). Sinon, on imprime simplement le caractère tapé au bon endroit dans FEN.

fonctions similaires : TRAIT_SAISIE_PARAM, SAISIE_1 ... SAISIE_12, TRAIT_ADM,TRAIT_ATT_REP, CREER_LISTE, SUPPRIMER_LISTE, AJOUT_ABONNE, SUPPRESS_ABONNE

5.4.10 INIT_OPTION1

entrées : OP1
ICON1_LABEL
ICON2_LABEL
...
ICON13_LABEL

pré-conditions : aucune

sorties : C101_PROC
C201_PROC
...
C1301_PROC

post-conditions : aucune

objectif : initialise la sous-fenêtre des icônes, en définissant pour chaque icône (ICONx_LABEL) la fonction (CxO1_PROC) qui est appelée quand cet icône est sélectionné avec la souris par l'utilisateur. Cette fonction place également les différents icônes dans la sous-fenêtre de manière à ce qu'ils soient centrés.

fonctions similaires : INIT_OPTION2

5.4.11 C301_PROC

entrées : PASS_COMMANDE

RES

DOC

pré-conditions : RES et DOC sont fermés

sorties : PASS_ICONE

PRESLIST

PERE_PRES_RES

PERE_PRES_DOC

FEN2_PIXWIN

post-conditions : PASS_ICONE est égal à 3, PERE_PRES_RES est égal à 0, PERE_PRES_DOC est égal à 0, PRESLIST est égal à 0

objectif : cette fonction est appelée quand l'utilisateur appuie sur le bouton du milieu ou de gauche de la souris alors que le curseur se trouve sur l'icône POUBELLE. Selon que l'utilisateur a auparavant selectionné la commande CONSULTER ou TRANSFERER (PASS_COMMANDE vaut 1 ou 2), ou aucune des deux (PASS_COMMANDE vaut 0), la fonction initialise la consultation de la collection de documents en POUBELLE en initialisant des témoins de présence de certains documents à 0 (PERE_PRES_RES, PERE_PRES-DOC, et PRESLIST) et en affichant le premier résumé de la collection des documents en POUBELLE dans la sous-fenêtre des résumés et le premier document dans la sous-fenêtre des documents, ou elle effectue le transfert (en affichant le message "transfert effectué") si le transfert est permis (c-à-d si le document

sélectionné provient d'une collection à partir de laquelle il est permis de transférer un document vers la collection des documents en POUBELLE).

fonctions similaires : C101_PROC, ..., C1301_PROC

5.4.12 C202_PROC

entrées : les témoins d'édition

pré-conditions : 1 témoin d'édition au maximum est à 1

sorties : les témoins d'édition

DOC_SELECT

post-conditions : 1 témoin d'édition au maximum est à 1, et c'est celui qui indique qu'une édition normale est en cours

objectif : cette fonction est appellée quand l'utilisateur appuye sur le bouton du milieu ou de gauche de la souris alors que le curseur se trouve sur la commande EDITER. Si aucune édition n'est en cours (tous les témoins d'édition sont à 0), on crée un processus dans lequel on exécute un "tool" qui va se placer sur la sous-fenêtre des résumés. Dans ce "tool", on exécute l'éditeur VI sur un fichier vide. On positionne le témoin d'édition normale en cours à 1. Sinon on imprime un message d'erreur dans la sous-fenêtre des messages (édition interdite pour le moment).

fonctions similaires : C102_PROC, C302_PROC, C402_PROC

6 MODE D'EMPLOI

6.1 INTRODUCTION

Nous allons d'abord présenter la façon dont on lance le programme. Ensuite, nous parlerons de l'accès à l'utilisation du courrier électronique (mot de passe). Après, nous donnerons les principes généraux d'utilisation des différentes fonctions. La possibilité d'exécuter plusieurs fonctions en parallèle sera aussi abordée. Nous poursuivrons en énumérant et en définissant les différentes commande-menus existantes. Et enfin, nous verrons le mode d'emploi par commande, et pour chaque commande, par icône.

6.2 LANCEMENT DU PROGRAMME

Pour lancer notre programme, il faut disposer des fichiers nécessaires à son exécution. Il y a tout d'abord le fichier contenant le programme lui-même, ensuite les fichiers contenant la représentation des icônes, et enfin les fichiers contenant les collections de documents.

Pour lancer un programme graphique, il faut être dans suntool. Pour notre programme, il faut donc taper SUNTOOL (on peut dès lors se servir de l'écran graphique) et ensuite, il faut taper le nom du programme : COURRIER. L'exécution du programme débute.

6.3 ACCES AU COURRIER ELECTRONIQUE

Une fois le programme lancé, pour pouvoir utiliser les fonctions du courrier électronique, il faut être un abonné à ce courrier et donc posséder le mot de passe. Le première chose qui sera dès lors demandée à l'utilisateur, sera le mot de passe. S'il est correct, il peut accéder aux fonctions du courrier électronique, sinon le programme se termine.

6.4 PRINCIPES GENERAUX D'UTILISATION

Les commandes et les icônes sont sélectionnés en déplaçant le curseur sur ceux-ci, et en enfonçant et relâchant le bouton central de la souris. Dans la suite du chapitre, nous dirons simplement "sélectionner une commande" et "sélectionner un icône" pour l'ensemble du mécanisme.

Pour activer une fonction, il faut, soit sélectionner une commande et un icône (ceci dans le cas des commandes CONSULTER et TRANSFERER), soit sélectionner une commande seule (dans le cas des commandes EDITER et QUITTER).

De même, les commande-menus sont sélectionnées en déplaçant le curseur dans la sous-fenêtre des documents ou des résumés (selon qu'on veut travailler sur les uns ou sur les autres), en enfonçant le bouton de droite de la souris pour faire apparaître

le menu, en déplaçant le curseur sur la commande-menu choisie et en relâchant le bouton de droite de la souris. Dans la suite du chapitre, nous parlerons simplement de "sélectionner une commande-menu".

Les commande-menus pour une consultation sont les mêmes pour les documents et pour les résumés de ces documents. Si elles sont sélectionnées dans la sous-fenêtre des documents, elles s'appliquent aux documents. Si elles sont sélectionnées dans la sous-fenêtre des résumés, elles s'appliquent aux résumés.

Chaque fois que le programme demande d'entrer des données à l'utilisateur, il le demande dans la sous-fenêtre des paramètres.

Lorsque le programme affiche un message pour l'utilisateur, il le fait toujours dans la sous-fenêtre des messages (à deux exceptions près qui seront mentionnées et justifiées plus loin). Dans la partie de droite de la sous-fenêtre des messages apparaitront toujours la commande et l'icône sélectionnés, pour que l'utilisateur sache ce qu'il est en train d'exécuter.

Lorsque l'utilisateur n'aura qu'un choix limité de lettres à donner comme valeur d'un paramètre (o ou n), le caractère tapé ne sera accepté que s'il fait partie du choix de solutions.

Lors d'une demande de paramètres, l'utilisateur peut annuler cette demande en appuyant sur la touche ‹esc›. A ce moment, la fonction dont dépendait la demande de paramètres est elle-même

annulée.

Des messages apparaitront lorsque ce sera nécessaire, pour confirmer, infirmer l'exécution de fonctions, et pour avertir l'utilisateur de ses erreurs. Tous ces messages ne seront pas repris en détail dans ce mode d'emploi.

Lors de l'exécution des commande-menus de la consultation bottin, l'utilisateur peut donner la valeur aux paramètres demandés sans devoir la taper, mais en déplaçant le curseur sur le nom de la liste ou le nom de l'abonné et en enfonçant et relâchant le bouton de gauche de la souris. Tout ceci pour essayer de faciliter au maximum le travail de l'utilisateur et pour essayer de minimiser le nombre d'erreurs.

6.5 POSSIBILITE DE TRAVAILLER EN PARALLELE

Ce point a déjà été abordé dans le chapitre 4 mais il est bon d'en reparler brièvement vu son importance.

Etant donné qu'il existe plusieurs sous-fenêtres avec pour chacune d'entre elles un rôle spécifique, deux fonctions n'utilisant pas les mêmes sous-fenêtres peuvent s'exécuter en parallèle. On peut par exemple commencer une consultation de la boite-in, et pendant qu'un document de la collection est à l'écran, faire un transfert de celui-ci vers la poubelle (cette opération ne requiert pas l'utilisation des sous-fenêtres des

documents et des résumés).

Lorsque l'utilisateur essaiera de lancer en parallèle deux fonctions dont l'exécution requiert la même sous-fenêtre, cette manoeuvre lui sera interdite et un message d'erreur l'en avertira.

6.6 DEFINITION DES COMMANDE-MENUS

- "PREMIER" : affiche à l'écran le premier document (résumé) de la collection à la place de celui qui s'y trouve.

- "DERNIER" : affiche à l'écran le dernier document (résumé) de la collection à la place de celui qui s'y trouve.

- "PRECEDENT" : affiche à l'écran le document (résumé) de la collection précédent celui qui s'y trouve. Le précédent du premier sera le dernier.

- "SUIVANT" : affiche à l'écran le document (résumé) de la collection suivant celui qui s'y trouve. Le suivant du dernier sera le premier.

- "SELECTION" : sélectionne le document (résumé) de la collection se trouvant à l'écran à ce moment.

- "MOD. CORPS" : entre dans l'éditeur de texte avec le document qui se trouve à ce moment à l'écran. La modification du

document se fait grâce aux commandes de l'éditeur VI et se termine lorsque l'utilisateur sort de l'éditeur.

- "TRAITEMENT" : active la fonction de consultation des documents en attente de traitement.

- "REPONSE" : active la fonction de consultation des documents en attente de réponse.

- "ACCUSE DE RECEPTION" : active la fonction de consultation des documents en attente d'accusé de réception.

- "MOD. REPONSE" : entre dans l'éditeur de texte avec le document qui est la réponse au document consulté en ce moment (la réponse peut être vide). La modification de la réponse se fait grâce aux commandes de l'éditeur VI et se termine lorsque l'utilisateur quitte l'éditeur.

- "RAPPEL" : entre dans l'éditeur de texte. L'utilisateur peut alors taper un rappel de demande de réponse. L'édition du rappel se fait grâce aux commandes de l'éditeur VI et se termine lorsque l'utilisateur quitte l'éditeur. Une fois l'éditeur quitté, les paramètres d'envoi du rappel sont demandés :

  titre :

  confidentiel (o|n) :

  recommandé (o|n) :

note :

mot-clés :

L'utilisateur peut taper cinq mot-clés. Il lui en sera demandés jusqu'à ce qu'il en ait tapé cinq ou jusqu'à ce qu'il ait tapé successivement deux fois RETURN. Lorsqu'il a donné une valeur à ceux-ci, ou bien lorsqu'il a annulé la demande de paramètres, l'exécution de la commande-menu se termine.

- "SUPPRIMER" : le document à l'écran à cet instant est supprimé.

- "MODIFIER" : entre dans l'éditeur de texte avec le document du signataire qui se trouve à ce moment à l'écran. La modification du document se fait grâce aux commandes de l'éditeur VI et se termine lorsque l'utilisateur quitte l'éditeur.

- "SIGNER" : le document à l'écran à cet instant est signé. Il peut dès lors sortir du signataire.

- "MOD. NOTE MODIF" : entre dans l'éditeur de texte avec la note de modification du document se trouvant à cet instant à l'écran (la note de modification peut être vide). L'utilisateur peut alors taper ou modifier cette note. L'édition de celle-ci se fait grâce aux commandes de l'éditeur VI et se termine lorsque l'utilisateur quitte

l'éditeur.

- "GESTION JOURNALIERE" : imprime dans la sous-fenêtre des paramètres l'ancienne date et demande la nouvelle à l'utilisateur :

  ancienne date : 6|6|1944
  nouvelle date :

  Lorsque l'utilisateur a répondu, la gestion journalière se termine.

- "STATISTIQUES" : demande à l'utilisateur :

  (a)bonné ou (r)apport général :

  - S'il veut des statistiques sur un seul abonné, il tape a. Le programme lui demande alors :

    identifiant d'abonné :
    date de début :
    date de fin :

    Lorsque l'utilisateur a donné une valeur à ces paramètres, les statistiques sur cet abonné sont affichées dans la sous-fenêtre des documents.
  - S'il veut un rapport général de statistique, il tape r. Le programme lui demande alors :

date de début :

date de fin :

Lorsque l'utilisateur a donné une valeur à ces paramètres, le rapport général de statistique est affiché dans la sous-fenêtre des documents.

- "CHANGEMENT MOT DE PASSE" : le programme demande à l'utilisateur :

nouveau mot de passe :

Lorsqu'il a répondu, on lui demande une seconde fois. S'il répond deux fois le même mot de passe, la fonction changement de mot de passe se termine, et le nouveau mot de passe est d'application. Dans le cas contraire, la fonction se termine, et l'ancien mot de passe reste d'application.

- "CHANGEMENT CLE DE CODAGE" : le programme demande à l'utilisateur :

clé de codage :

- S'il donne la bonne clé de codage, on lui demande :

nouvelle clé de codage :

Lorsqu'il a répondu, on lui demande une seconde fois. S'il répond deux fois la même clé, la fonction changement de clé de codage se termine, et la nouvelle clé est d'application. Dans le cas contraire, la fonction se termine, et l'ancienne clé reste d'application.

- S'il donne une mauvaise clé de codage, la fonction se termine.

- "CREATION D'UN ABONNE" : le programme demande à l'utilisateur :

(s)ervice, (p)atron, (se)crétaire, (d)actylo :

- S'il veut créer un service, il tape s. Deux autres paramètres lui sont demandés :

identifiant du service :

identifiant du patron du service :

Lorsqu'il a donné une valeur à ces paramètres, le nouveau service est créé et la fonction se termine.

- S'il veut créer un nouveau patron, il tape p. Deux autres paramètres lui sont demandés :

identifiant du patron :

identifiant du service de ce patron :

Lorsqu'il a donné une valeur à ces paramètres, le nouveau patron est créé et la fonction se termine.

- S'il veut créer une dactylo, il tape d. Deux autres paramètres lui sont demandés :

  identifiant de la dactylo :

  identifiant du service :

  Lorsqu'il a donné une valeur à ces paramètres, la nouvelle dactylo est créée et la fonction se termine.

- S'il veut créer une secrétaire, il tape se. Deux autres paramètres lui sont demandés :

  identifiant de la secrétaire :

  identifiant du service :

  Lorsqu'il a donné une valeur à ces paramètres, la nouvelle secrétaire est créée et la fonction se termine.

- "SUPPRESSION D'UN ABONNE" : le programme demande à l'utilisateur :

  identifiant :

Lorqu'il a donné l'identifiant, on lui demande une

confirmation. S'il confirme, l'abonné est supprimé et la fonction se termine. Dans le cas contraire, la fonction se termine sans suppression de l'abonné.

- "MODIFICATION D'UN ABONNE" : le programme demande à l'utilisateur :

  (s)ervice, (se)crétaire, (d)actylo :

  - S'il veut modifier un service (en lui donnant un nouveau patron), il tape s. Deux autres paramètres lui sont demandés :

    identifiant du service :
    identifiant du nouveau patron du service :

    Lorsqu'il a donné une valeur à ces paramètres, le service est modifié et la fonction se termine.

  - S'il veut changer une secrétaire de service, il tape se. Un autre paramètre lui est demandé :

    identifiant du nouveau service :

    Lorsqu'il a donné une valeur à ce paramètre, la secrétaire a changé de service et la fonction se termine.

- S'il veut changer une dactylo de service, il tape d. Un autre paramètre lui est demandé :

  identifiant du nouveau service :

  Lorsqu'il a donné une valeur à ce paramètre, la dactylo a changé de service et la fonction se termine.

- "DESCRIPTION D'UN ABONNE" : le programme demande à l'utilisateur :

  identifiant :

  Lorsqu'il a donné une valeur à ce paramètre, la description de l'abonné apparait dans la sous-fenêtre des documents et la fonction se termine.

- "CREER LISTE" : le programme demande à l'utilisateur :

  nom de la liste :

  Le programme lui demande cette donnée tant qu'il entre un nom de liste qui existe déjà. Ensuite, un nouveau paramètre lui est demandé :

  nom d'abonné :

Des noms d'abonnés constituant cette liste lui sont demandés jusqu'à ce qu'il tape successivement deux fois RETURN. A ce moment, la fonction se termine. Les noms d'abonnés n'existant pas dans la liste des abonnés sont refusés.

- "SUPPR.LISTE" : le programme demande à l'utilisateur :

  nom de la liste :

  Le programme lui demande cette donnée tant qu'il entre un nom de liste qui n'existe pas. Lorsqu'il tape un nom de liste existant (sauf le nom de la liste des abonnés que seul l'administrateur peut supprimer), la liste est supprimée et la fonction se termine.

- "AJOUT. ABONNE" : le programme demande à l'utilisateur :

  nom de la liste :

  L'utilisateur doit alors taper un nom de liste qui n'existe pas et qui est différent de celui de la liste des abonnés (en effet, modifier la liste des abonnés est une fonction réservée à l'administrateur). On lui demande le nom de la liste tant que celui-ci n'est pas valide. Lorsqu'il a tapé un nom de liste valide, un autre paramètre lui est demandé :

  nom d'abonné :

Les abonnés doivent faire partie de la liste des abonnés. La fonction se termine lorsque l'utilisateur tape successivement deux fois RETURN.

- "SUPPR.ABONNE" : le programme demande à l'utilisateur :

  nom de la liste :

  Le programme lui demande cette donnée tant qu'il entre un nom de liste qui n'existe pas ou qu'il entre "liste des abonnés" (en effet, pour pouvoir modifier la liste des abonnés, il faut donner le mot de passe d'administration). Lorsqu'il a tapé un nom de liste valide, un autre paramètre lui est demandé :

  nom d'abonné :

  Les abonnés doivent faire partie de la liste tapée. La fonction se termine lorsque l'utilisateur tape successivement deux fois RETURN.

6.7 LES CONSULTATIONS

6.7.1 CONSULTATION DE LA POUBELLE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône POUBELLE. Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre

respective. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION.

6.7.2 CONSULTATION DES ARCHIVES

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône ARCHIVES. Le programme lui demande alors :

document (e)nvoyé ou (r)eçu :

- S'il veut consulter un document archivé qu'il a envoyé, il tape e suivi de RETURN. Ceci entraine une nouvelle demande de paramètres.

identifiant (o|n) :

- S'il ne veut consulter qu'un document et qu'il connait le numéro d'envoi et la date d'envoi de celui-ci, il tape o suivi de RETURN et ces deux nouveaux paramètres lui sont demandés.

date d'envoi :
numéro d'envoi :

Lorsqu'il a répondu à ces deux paramètres, le document correspondant à ce qu'il a demandé apparait dans la sous-fenêtre des documents. Plusieurs commande-menus peuvent alors être sélectionnées : SELECTION, MOD.

CORPS.

- S'il veut consulter une liste de documents, il tape n suivi de RETURN et trois nouveaux paramètres lui sont demandés.

  date d'envoi :

  destinataire :

  mots-clés :

  Pour les mots-clés, on lui demande d'en donner un autre, jusqu'à ce qu'il en ait tapé cinq ou qu'il ait tapé deux fois RETURN de suite. Lorsqu'il a donné tous ces paramètres, le premier document d'une liste de documents correspondant aux paramètres entrés s'affiche dans la sous-fenêtre des documents. Le résumé correspondant s'affiche dans la sous-fenêtre des résumés. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION, MOD. CORPS.

- S'il veut consulter un document archivé qu'il a reçu, il tape r suivi de RETURN. Ceci entraine une nouvelle demande de paramètres.

  identifiant (o|n) :

- S'il ne veut consulter qu'un document et qu'il connait

le numéro d'envoi, la date d'envoi et l'expéditeur de celui-ci, il tape o suivi de RETURN et ces trois nouveaux paramètres lui sont demandés.

expéditeur :
date d'envoi :
numéro :

Lorsqu'il a répondu à ces trois paramètres, le document correspondant à ce qu'il a demandé apparait dans la sous-fenêtre des documents. Plusieurs commande-menus peuvent alors être sélectionnées : SELECTION, MOD. CORPS.

- S'il veut consulter une liste de documents, il tape n suivi de RETURN et trois nouveaux paramètres lui sont demandés.

date de réception :
expéditeur :
mots-clés :

Pour les mots-clés, on lui demande d'en donner un autre, jusqu'à ce qu'il en ait tapé cinq ou qu'il ait tapé deux fois RETURN de suite. Lorsqu'il a donné tous ces paramètres, le premier document d'une liste de documents correspondant aux paramètres entrés, s'affiche dans la sous-fenêtre des documents. Le résumé correspondant

s'affiche dans la sous-fenêtre des résumés. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION, MOD. CORPS.

6.7.3 CONSULTATION DES DOCUMENTS EN ATTENTE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône DOCUMENTS EN ATTENTE. Un menu est alors accessible à celui-ci dans la sous-fenêtre des résumés. L'utilisateur en est prévenu par le message "veuillez activer le menu avec le bouton de droite de la souris" qui apparait dans la sous-fenêtre des résumés. Dans ce menu, il y a trois commande-menus : TRAITEMENT, REPONSE, ACCUSE DE RECEPTION.

6.7.3.1 CONSULTATION DES DOCUMENTS EN ATTENTE DE TRAITEMENT

Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective. Plusieurs commande-menus peuvent alors être sélectionnées :PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION, MOD. CORPS, MOD. REPONSE.

6.7.3.2 CONSULTATION DES DOCUMENTS EN ATTENTE DE REPONSE

Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective.

Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION, RAPPEL

6.7.3.3 CONSULTATION DES DOCUMENTS EN ATTENTE D'UN ACCUSE DE RECEPTION

Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION.

6.7.4 CONSULTATION DE LA BOITE-IN

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône BOITE-IN. Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION.

6.7.5 CONSULTATION DU SIGNATAIRE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône SIGNATAIRE. Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective. A ce moment , deux menus sont disponibles. Lorsque, le curseur étant dans la sous-fenêtre des documents (résumés), on enfonce le bouton de droite de la souris, deux menus apparaissent. L'un de ceux-ci est superposé à l'autre dont on ne voit que le

titre. Pour changer l'ordre de superposition des menus, on déplace le curseur sur le titre du menu caché, on enfonce le bouton de gauche de la souris, et on le relâche ensuite (tout cela en maintenant enfoncé le bouton de droite). On ne sait sélectionner que les commande-menus du menu visible. Cela se fait comme pour toutes les autres commande-menus.

Le premier menu est un menu de consultation. Les commande-menus suivantes peuvent y être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION.

Le second menu est un menu de modification. Les commande-menus suivantes peuvent y être sélestionnées : SUPPRIMER, MODIFIER, SIGNER, MOD. NOTE MODIF.

6.7.6 CONSULTATION DE LA POSTE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône POSTE. Si l'utilisateur avait sélectionné un document, la date de réception de ce document lui sera donnée dans la sous-fenêtre des messages. S'il n'avait pas sélectionné un document, le message "pas de document sélectionné" apparait.

6.7.7 CONSULTATION MOT DE PASSE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône MOT DE PASSE. Le programme lui demande alors :

mot de passe d'abonné :

L'utilisateur doit alors donner son mot de passe d'abonné. Il a droit à cinq tentatives. Si, après celles-ci, il n'a toujours pas donné le bon mot de passe, la consultation mot de passe se termine.

S'il a donné le mot de passe correct, on lui demande alors :

nouveau mot de passe d'abonné :

Lorsqu'il a donné son nouveau mot de passe, on le lui redemande une seconde fois. Si les deux nouveaux mots de passe sont les mêmes, le même processus s'engage pour le mot de passe de confidentialité. S'ils ne sont pas les mêmes, on les redemande tous les deux (et ceci jusqu'à ce qu'ils correspondent).

Si l'utilisateur veut changer seulement son mot de passe de confidentialité, il doit taper successivement deux fois RETURN lorqu'on lui demande son mot de passe d'abonné. Cela signifie qu'il ne change pas ce dernier.

6.7.8 CONSULTATION DES REPONSES RECUES

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône BOITE-IN. Si aucun document n'a été sélectionné, le message "PAS DE DOCUMENT SELECTIONNE" est affiché, et la

consultation est terminée. Si, au contraire, une sélection a été effectuée, le premier document (et son résumé) de la collection des réponses reçues à ce document sélectionné apparait alors, chacun dans leur sous-fenêtre respective. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION, MOD. CORPS.

6.7.9 CONSULTATION DU DOCUMENT PERE

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône DOCUMENT PERE. Le premier document de la collection et son résumé apparaissent alors chacun dans leur sous-fenêtre respective. Plusieurs commande-menus peuvent alors être sélectionnées : PREMIER, DERNIER, PRECEDENT, SUIVANT, SELECTION.

6.7.10 CONSULTATION DE L'ADMINISTRATION

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône ADMINISTRATION. Le programme demande alors

mot de passe d'administration :

Si l'utilisateur tape un mot de passe erroné, la consultation de l'administration se termine. Si le bon mot de passe d'administration est donné, un menu est accessible à l'utilisateur dans la sous-fenêtre des résumés. L'utilisateur en est prévenu par le message "veuillez activer le menu avec le bouton de droite de

la souris" qui apparait dans la sous-fenêtre des résumés. Dans ce menu, il y a huit commande-menus : GESTION JOURNALIERE, STATISTIQUES, CHANGEMENT MOT DE PASSE, CHANGEMENT CLE DE CODAGE, CREATION D'UN ABONNE, SUPPRESSION D'UN ABONNE, MODIFICATION D'UN ABONNE, DESCRIPTION D'UN ABONNE.

6.7.11 CONSULTATION DU BOTTIN

L'utilisateur doit sélectionner la commande CONSULTER puis l'icône BOTTIN. La liste des listes d'abonnés apparait alors dans la sous-fenêtre des documents, et les commande-menus suivantes sont disponibles : CREER LISTE, SUPPR. LISTE, AJOUT. ABONNE, SUPPR. ABONNE.

On peut obtenir les abonnés d'une liste en déplaçant le curseur sur cette liste et en enfonçant et relâchant le bouton central de la souris.

6.8 TRANSFERT

6.8.1 TRANSFERT VERS LA BOITE-OUT

L'utilisateur doit sélectionner la commande TRANSFERT puis l'icône BOITE-OUT. Les paramètres suivant lui sont alors demandés :

nom du destinataire :

titre du document :

rédacteur :

confidentiel (o|n) :

rappel (o|n) :

recommandé (o|n) :

réponse (o|n) :

mot-clés :

S'il répond o au paramètre rappel, un nouveau paramètre lui est demandé :

dem. de réponse (o|n) :

S'il répond o à ce paramètre, trois nouveaux paramètres lui sont demandés :

nom d'expéditeur :

date d'envoi :

numro :

Lorsqu'il a répondu à tous ces paramètres, la fonction se termine et un message annonce que le document a été envoyé.

6.8.2 TRANSFERT VERS L'IMPRIMANTE

L'utilisateur doit sélectionner la commande TRANSFERT puis l'icône IMPRIMANTE.

Si un document a été sélectionné, celui-ci sort sur l'imprimante. Si aucun document n'a été sélectionné, la fonction se termine et un message annonce qu'aucun document n'a été sélectionné.

6.8.3 LES AUTRES TRANSFERTS

L'utilisateur doit sélectionner la commande TRANSFERT puis un icône parmi les suivants : POUBELLE, ARCHIVES, DOCUMENTS EN ATTENTE, SIGNATAIRE, REPONSE RECUES.

Si un document a été sélectionné, la fonction se termine et un message annonce à l'utilisateur que le transfert a été effectué. Si aucun document n'a été sélectionné, la fonction se termine et un message annonce qu'aucun document n'a été sélectionné.

6.9 EDITION

L'utilisateur doit sélectionner la commande EDITER. L'éditeur de texte apparait alors sur la sous-fenêtre des résumés.

A ce moment, il n'y a pas de texte. L'utilisateur peut alors créer son document grâce aux commandes de l'éditeur VI. L'édition se termine lorsque l'utilisateur sort de l'éditeur. Le document qu'il a tapé est alors transféré automatiquement dans la collection des documents en attente de traitement.

6.10 SORTIE DU PROGRAMME

Pour sortir du programme, l'utilisateur doit sélectionner la commande QUITTER. A ce moment, apparait un message demandant à celui-ci de confirmer en appuyant sur le bouton de gauche de la souris, ou d'infirmer en appuyant sur un des deux autres boutons. S'il confirme, l'exécution du programme se termine. S'il infirme, le programme poursuit son exécution.

7 CONCLUSION

Ce mémoire consistant en l'amélioration de l'interface d'un logiciel de courrier électronique existant, nous avons donc, dans un premier temps, rappelé les fonctions offertes par ce logiciel et les concepts qui s'y rattachaient.

Pour pouvoir améliorer l'interface de ce logiciel, nous nous sommes efforcés, dans un deuxième temps, de savoir ce qu'était un bon interface. Nous avons donc proposé une série de qualités qu'un bon interface doit respecter et des recommandations à suivre dans la conception. Ces qualités et recommandations avaient en grande partie pour but de faciliter au maximum la tâche de l'utilisateur.

Mettre au point un interface répondant aux qualités nommées ne pouvait se faire que sur un ordinateur possédant des caractéristiques particulières, en l'occurrence le SUN, avec notamment son écran de grande taille, sa souris, et la possibilité d'utiliser la programmation des fenêtres.

Disposant donc d'un sujet, d'une base théorique et d'un matériel adéquat, nous avons développé un interface en essayant au maximum que celui-ci respecte les qualités requises. Nous avons voulu qu'il soit simple à utiliser et possédant un mode d'emploi facile à comprendre étant donné qu'il était destiné à des utilisateurs avec peu d'expérience.

Cependant il existe des limites à ce mémoire : premièrement, il a été conçu pour des utilisateurs peu expérimentés. Un interface de ce même logiciel pourrait être très différent, si il était destiné à un type d'utilisateur avec un niveau d'expérience plus élevé. En effet, le SUN permet de faire des interfaces plus puissants que celui que nous avons réalisé, mais cette puissance d'utilisation entraine une plus grande difficulté d'utilisation, surtout pour une personne qui n'est pas habituée à ce type d'interface. La meilleure solution consisterait à réaliser un interface à complexité variable, en fonction de l'expérience de l'utilisateur.

Deuxièmement, l'interface a été réalisé pour un logiciel de courrier électronique, mais indépendamment de celui-ci en ce qui concerne le matériel et le langage de programmation utilisés. Lier les deux programmes pour pouvoir s'en servir de façon opérationnelle demanderait à notre avis un travail assez conséquent.

Troisièmement notre interface a été testé uniquement par nous-mêmes alors qu'il est destiné à des personnes peu expérimentées. Certaines choses peuvent nous avoir semblé évidentes alors qu'elles ne le sont peu-être pas pour tout le monde.

BIBLIOGRAPHIE

- Lee A., Woo C. C., Lochovsky F. H., "Office aid : An Integrated Document Management System", Communications of the ACM, pp. 170-180, 1984.

- Tesler L., "The Smalltalk Environment", BYTE vol. 6, No 8, pp. 90-147, 1981.

- Smith D. C., Irby C., Kimball R., Verplank B., Harslem E., "Designing the Star User Interface", BYTE vol. 7, No 4, pp. 242-282, 1982.

- Woo C. C., Lochovsky F. H., "A System for Interactively Designing Message Templates", Proceedings of the IEEE COMPCON Conference, 1983.

- Lee A., Lochovsky F. H., "Enhancing the Usability of an Office Information System Through Direct Manipulation", Proceedings CHI Conference on Human Factors in Computing Systems, December 1983.

- O. T. Systems Conceptual Functional Requirements, pp. 5.1-5.7 & 6.5-6.9, July 1980.

- Scapin D. L., "Guide Ergonomique de Conception des Interfaces Homme-Ordinateur", Institut National de Recherche en Informatique et en Automatique.

- Josis A., Bernard P., "Courrier électronique", mémoire des FUNDP, institut d'informatique, 1985.

- The Unix System, SUN Technical Report,1985.

- Sunwindows Programmer´s Guide, SUN Technical Report, January 1984.

- Sunwindows Reference Manual, Sun Technical Report, January 1984.

- Kernighan B. W., Ritchie D. M., "The C Programming Language", Prentice-Hall Inc., 1978.

ANNEXE : LISTING COMMENTE DU PROGRAMME SOURCE

```
/*****************************************************************************/
/*********************  DEBUT DU PROGRAMME  **********************************/
/*****************************************************************************/

#include <suntool/tool_hs.h>
#include <suntool/ttysw.h>
#include <stdio.h>

/** tool **/
static struct tool *tool;

/** sous-fenetre simulateur de terminal **/
static struct toolsw *tty;

/** fonctions de gestion des signaux de terminaison de processus et de **/
/** changement de taille ou de position du tool                        **/
static sigc(),sigw();

/** tableau contenant les arguments de la commande que l'on va executer **/
/** dans la sous-fenetre de simulation de terminal                      **/
static char *argv2[] = {
                         "                \0",
                         "                \0",
                         "                \0",
                        };

/*****************************************************************************/

main(argc,argv)
int argc;
char **argv;
{
struct rect rr;

/** si le deuxieme argument de la ligne de commande est egal a 3 -> **/
/** construction d'un rectangle de dimension de fen3                **/
      if (*(*(argv + 2)) == '3') {
         rect_construct(&rr,5,206,570,600);
      }
      else {

/** si le deuxieme argument de la ligne de commande n'est pas egal **/
/** a 3 -> il est egal a 2, construction d'un rectangle de         **/
/** dimension de fen2                                              **/
           rect_construct(&rr,581,206,565,600);
      }

/** creation du tool **/
      if ((tool = tool_create("",NULL,&rr,(struct icon *)NULL))
          == (struct tool *)NULL)
      exit(1);

/** creation de la sous-fenetre de simulation de terminal **/
      if ((tty = ttysw_createtoolsubwindow(tool,"",TOOL_SWEXTENDTOEDGE,
           TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
      exit(1);

/** gestion du signal de changement de taille ou de position de **/
/** la fenetre                                                  **/
      signal(SIGWINCH,sigw);
```

```
/** installation du tool **/
        tool_install(tool);

/** gestion du signal de fin de processus **/
        signal(SIGCHLD,sigc);

/** on remplit le tableau avec la commande que l'on veut effectuer **/
/** dans la sous-fenetre, c-a-d un 'vi' du fichier dont le nom est **/
/** le deuxieme argument de la ligne de commande                   **/
        *argv2 = "vi\0";
        *(argv2+1) = *(argv+1);
        *(argv2+2) = NULL;

/** creation d'un processus qui va tourner dans la sous-fenetre de **/
/** simulation de terminal, processus qui consiste en l'execution  **/
/** de l'editeur de texte 'vi'                                     **/
        ttysw_fork(tty->ts_data,argv2,
                   &tty->ts_io.tio_inputmask,
                   &tty->ts_io.tio_outputmask,
                   &tty->ts_io.tio_exceptmask);

/** positionnement de la souris **/
        win_setmouseposition(tty->ts_windowfd,100,100);

/** entree dans la boucle principale d'un tool **/
        tool_select(tool,1);

/** destruction du tool **/
        tool_destroy(tool);
        exit(0);

}

/*************************************************************************/

/**** gestion des signaux de fin de processus ****/

static sigc()
{
        tool_sigchld(tool);
}

/*************************************************************************/

/**** gestion des signaux de modification de la fenetre ****/

static sigw()
{
        tool_sigwinch(tool);
}

/*************************************************************************/
/*********************  FIN DU PROGRAMME  ********************************/
/*************************************************************************/
```

```
/*****************************************************************/
/*****************************************************************/
/**********************  DEBUT DU PROGRAMME  *********************/
/*****************************************************************/
/*****************************************************************/

/*****************************************************************/
/**********************  FICHIERS  A INCLURE  ********************/
/*****************************************************************/

#include <suntool/tool_hs.h>
#include <suntool/msgsw.h>
#include <suntool/optionsw.h>
#include <suntool/menu.h>
#include <stdio.h>
#include "icones/courrier.icon"      /**  ICONE GENERAL **/
#include "icones/out.icon"           /**  ICONE BOITE AUX LETTRES  **/
#include "icones/bottin.icon"        /**  ICONE BOTTIN  **/
#include "icones/poubelle.icon"      /**  ICONE POUBELLE **/
#include "icones/archive.icon"       /**  ICONE ARCHIVES **/
#include "icones/doc_at_trait.icon"/**  ICONE DOCUMENTS EN ATTENTE DE **/
                                     /**  TRAITEMENT                    **/
#include "icones/printer.icon"       /**  ICONE IMPRIMANTE **/
#include "icones/in.icon"            /**  ICONE BOITE IN **/
#include "icones/signataire.icon"    /**  ICONE SIGNATAIRE **/
#include "icones/poste.icon"         /**  ICONE POSTE **/
#include "icones/mot.icon"           /**  ICONE MOT **/
#include "icones/rep_rec.icon"       /**  ICONE REPONSES RECUES A UN **/
                                     /**  DOCUMENT ENVOYE            **/
#include "icones/pere.icon"          /**  ICONE PERE **/
#include "icones/adm.icon"           /**  ICONE ADMINISTRATION **/

/*****************************************************************/
/**********************   DECLARATIONS   *************************/
/*****************************************************************/

extern struct pixfont *pw_pfsysopen();
static struct pixfont *font;

/** zone memoire pour les icones **/
mpr_static(icon_cour,64,64,1,icon_data);
mpr_static(icon_1,64,64,1,icon1_data);
mpr_static(icon_2,64,64,1,icon2_data);
mpr_static(icon_3,64,64,1,icon3_data);
mpr_static(icon_4,64,64,1,icon4_data);
mpr_static(icon_5,64,64,1,icon5_data);
mpr_static(icon_6,64,64,1,icon6_data);
mpr_static(icon_7,64,64,1,icon7_data);
mpr_static(icon_8,64,64,1,icon8_data);
mpr_static(icon_9,64,64,1,icon9_data);
mpr_static(icon_10,64,64,1,icon10_data);
mpr_static(icon_11,64,64,1,icon11_data);
mpr_static(icon_12,64,64,1,icon12_data);
mpr_static(icon_13,64,64,1,icon13_data);

/** definition de l'icone courrier **/
static struct icon icon = {TOOL_ICONWIDTH,TOOL_ICONHEIGHT,
                           (struct pixrect *)NULL,
                           {0,0,TOOL_ICONWIDTH,TOOL_ICONHEIGHT},
                           &icon_cour,{0,0,0,0},(char *)NULL,
                           (struct pixfont *)NULL,0};
```

```
/** objets de l'optionsubwindow des icones **/
static struct typed_pair icon1_label = {IM_GRAPHIC,(caddr_t)&icon_1};
static struct typed_pair icon2_label = {IM_GRAPHIC,(caddr_t)&icon_2};
static struct typed_pair icon3_label = {IM_GRAPHIC,(caddr_t)&icon_3};
static struct typed_pair icon4_label = {IM_GRAPHIC,(caddr_t)&icon_4};
static struct typed_pair icon5_label = {IM_GRAPHIC,(caddr_t)&icon_5};
static struct typed_pair icon6_label = {IM_GRAPHIC,(caddr_t)&icon_6};
static struct typed_pair icon7_label = {IM_GRAPHIC,(caddr_t)&icon_7};
static struct typed_pair icon8_label = {IM_GRAPHIC,(caddr_t)&icon_8};
static struct typed_pair icon9_label = {IM_GRAPHIC,(caddr_t)&icon_9};
static struct typed_pair icon10_label = {IM_GRAPHIC,(caddr_t)&icon_10};
static struct typed_pair icon11_label = {IM_GRAPHIC,(caddr_t)&icon_11};
static struct typed_pair icon12_label = {IM_GRAPHIC,(caddr_t)&icon_12};
static struct typed_pair icon13_label = {IM_GRAPHIC,(caddr_t)&icon_13};
static caddr_t c1o1_item,c2o1_item,c3o1_item,c4o1_item,c5o1_item;
static caddr_t c6o1_item,c7o1_item,c8o1_item,c9o1_item,c10o1_item;
static caddr_t c11o1_item,c12o1_item,c13o1_item;

/** tool et subwindows **/
static struct tool *tool1;
static struct toolsw *fen_mes,*fen,*fen2,*fen3,*op1,*op2;

/** fonctions **/
static sigw(),sigc(),fen_sighandler(),init_fen(),fen2_sighandler();
static fen_input(),fen2_input(),init_fen2(),fin_fen_input();
static imprim_liste(),fen_mes_input(),init_fen_mes(),fen3_sighandler();
static fen_mes_sighandler(),titre();
static init_option2(),c1o2_proc(),c2o2_proc(),c3o2_proc(),c4o2_proc();
static init_option1(),c1o1_proc(),c2o1_proc(),imprim_char_suiv_fen();
static exist_abonne(),exist_liste(),trait_att_rep();
static init_pass_commande(),init_pass_icone();
static init_tab_cont_fen2(),put_tab_cont_fen2(),get_tab_cont_fen2();
static creer_liste(),supprimer_liste(),ajout_abonne(),suppress_abonne();
static saisie_1(),saisie_2_3(),saisie_4(),saisie_5(),saisie_6();
static saisie_7(),saisie_8(),saisie_9(),saisie_10(),saisie_11();
static imprim_clic(),c5o1_proc(),c6o1_proc(),c7o1_proc(),c8o1_proc();
static fen3_sighandler(),fen3_input(),c3o1_proc(),c4o1_proc();
static c9o1_proc(),c10o1_proc(),saisie_12(),c11o1_proc(),c12o1_proc();
static imprime(),ges_menu_cons_res(),ges_menu_cons_doc();
static put_perm(),get_perm(),trait_mot_passe(),ges_menu_attente();
static trait_archive(),suite_archive(),c13o1_proc();

/** pixwin des 4 windows non-standards **/
static struct pixwin *fen_pixwin,*fen2_pixwin,*fen3_pixwin;
static struct pixwin *fen_mes_pixwin;

/** objets de l'optionsubwindow des commandes generales **/
static struct typed_pair c1o2 = {IM_TEXT,"CONSULTER"};
static struct typed_pair c2o2 = {IM_TEXT,"EDITER"};
static struct typed_pair c3o2 = {IM_TEXT,"TRANSFERER"};
static struct typed_pair c4o2 = {IM_TEXT,"QUITTER"};
static caddr_t osw, c1o2_item, c2o2_item, c3o2_item, c4o2_item;

/** buffer contenant les messages pour la messagesubwindow **/
static char buf[256];

/** longueurs des tableaux contenant les listes **/
static int lg_tabbot = 4;
static int lg_tababbe = 4;
static int lg_tabprof =4;
```

```
static int lg_tabelev =4;
static int lg_taball =15;

/** tableaux contenant les listes **/
static char *tabbot[] = {
                         "liste des abonnes\0                    ",
                         "liste des abbes\0                      ",
                         "liste des profs\0                      ",
                         "liste des eleves\0                     ",
                         "                                     \0",
                         "                                     \0"
                         };
static char *tababbe[] = {
                          "l'abbe resina\0                       ",
                          "l'abbe cane\0                         ",
                          "l'abbe chamelle\0                     ",
                          "l'abbe tise\0                         ",
                          "                                     \0",
                          "                                     \0"
                         };
static char *tabprof[] = {
                          "bodart\0                              ",
                          "van bastelaer\0                       ",
                          "van lamsweerde\0                      ",
                          "brunin\0                              ",
                          "                                     \0",
                          "                                     \0"
                         };
static char *tabelev[] = {
                           "gonay\0                                ",
                           "henry\0                                ",
                           "hody\0                                 ",
                           "de crane\0                             ",
                           "                                      \0",
                           "                                      \0"
                          };
static char *taball[] = {
                          "l'abbe tise\0",
                          "l'abbe resina\0",
                          "l'abbe chamelle\0",
                          "l'abbe cane\0",
                          "bodart\0",
                          "van bastelaer\0",
                          "van lamsweerde\0",
                          "brunin\0",
                          "gonay\0",
                          "henry\0",
                          "hody\0",
                          "de crane\0",
                          "kimus\0",
                          "l'abbe tonniere\0",
                          "dunon\0",
                          "                                      \0"
                         };

/** fonction d'impression des pop-up menus **/
extern struct menuitem *menu_display();

/** evenement correspondant aux inputs des windows fen et fen2 **/
static struct inputevent ievent_fen2,ievent_fen,ievent_fen3;
static struct inputevent ievent_fen_mes;
```

```
/** temoin des commandes **/
static int pass_commande = 0;

/** temoin des icones **/
static int pass_icone = -1;

/** menu de choix de l'attente **/
static struct menuitem *menu_attente;
struct menuitem menu_attente_item[] = {
        {MENU_IMAGESTRING, "traitement",          (caddr_t)'t'},
        {MENU_IMAGESTRING, "reponse",             (caddr_t)'r'},
        {MENU_IMAGESTRING, "accuse de reception", (caddr_t)'a'},
                                                  };
static struct menu menu_attente_body = {
        MENU_IMAGESTRING, "Attente de",sizeof(menu_attente_item)
        / sizeof(struct menuitem),menu_attente_item,
        (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_attente_ptr = &menu_attente_body;
static char choix_attente = 'x';

/** menu de la consultation du bottin **/
static struct menuitem *menu1_fen2;
struct menuitem menu1_fen2_item[] = {
        {MENU_IMAGESTRING, "creer liste",     (caddr_t)'c'},
        {MENU_IMAGESTRING, "suppr. liste",    (caddr_t)'s'},
        {MENU_IMAGESTRING, "ajout. abonne",   (caddr_t)'a'},
        {MENU_IMAGESTRING, "suppr. abonne",   (caddr_t)'d'},
                                              };
static struct menu menu1_fen2_body = {
        MENU_IMAGESTRING, "Gestion listes",sizeof(menu1_fen2_item)
        / sizeof(struct menuitem),menu1_fen2_item,
        (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu1_fen2_ptr = &menu1_fen2_body;

/** tableau des commandes executees dans la ttysubwindow **/
static char *argv2[] = {
                        "                                ",
                        "                                ",
                        "                                "
                       };

/** temoin du choix du menu de la consultation du bottin **/
static char choice_menu1_fen2 = 'x';

/** menu des consultations **/
static struct menuitem *menu_cons;
struct menuitem menu_cons_item[] = {
         {MENU_IMAGESTRING, "premier",    (caddr_t)'p'},
         {MENU_IMAGESTRING, "dernier",    (caddr_t)'d'},
         {MENU_IMAGESTRING, "precedent",  (caddr_t)'a'},
         {MENU_IMAGESTRING, "suivant",    (caddr_t)'v'},
         {MENU_IMAGESTRING, "selection",  (caddr_t)'s'},
                                        };
static struct menu menu_cons_body = {
        MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_item)
        / sizeof(struct menuitem),menu_cons_item,
        (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_cons_ptr = &menu_cons_body;

/** temoin du choix du menu des consultations **/
```

```
static char choice_menu_cons = 'x';

/** temoin du resume courant **/
static int numres = 1;

/** temoin du document courant **/
static int numdoc = 1;

/** menu des consultations des documents en attente de reponse **/
static struct menuitem *menu_cons_att;
struct menuitem menu_cons_att_item[] = {
        {MENU_IMAGESTRING, "premier",    (caddr_t)'p'},
        {MENU_IMAGESTRING, "dernier",    (caddr_t)'d'},
        {MENU_IMAGESTRING, "precedent", (caddr_t)'a'},
        {MENU_IMAGESTRING, "suivant",    (caddr_t)'v'},
        {MENU_IMAGESTRING, "selection", (caddr_t)'s'},
        {MENU_IMAGESTRING, "rappel",     (caddr_t)'r'},
                                        };
static struct menu menu_cons_att_body = {
       MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_att_item)
       / sizeof(struct menuitem),menu_cons_att_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_cons_att_ptr = &menu_cons_att_body;

/** menu de la consultation des documents en attente de traitement **/
static struct menuitem *menu_cons_att_tr;
struct menuitem menu_cons_att_tr_item[] = {
        {MENU_IMAGESTRING, "premier",        (caddr_t)'p'},
        {MENU_IMAGESTRING, "dernier",        (caddr_t)'d'},
        {MENU_IMAGESTRING, "precedent",      (caddr_t)'a'},
        {MENU_IMAGESTRING, "suivant",        (caddr_t)'v'},
        {MENU_IMAGESTRING, "selection",      (caddr_t)'s'},
        {MENU_IMAGESTRING, "mod. corps",     (caddr_t)'z'},
        {MENU_IMAGESTRING, "mod. reponse",   (caddr_t)'w'},
                                        };
static struct menu menu_cons_att_tr_body = {
       MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_att_tr_item)
       / sizeof(struct menuitem),menu_cons_att_tr_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_cons_att_tr_ptr = &menu_cons_att_tr_body;

/** menu de la consultation des reponses recues **/
static struct menuitem *menu_cons_rep_rec;
struct menuitem menu_cons_rep_rec_item[] = {
        {MENU_IMAGESTRING, "premier",        (caddr_t)'p'},
        {MENU_IMAGESTRING, "dernier",        (caddr_t)'d'},
        {MENU_IMAGESTRING, "precedent",      (caddr_t)'a'},
        {MENU_IMAGESTRING, "suivant",        (caddr_t)'v'},
        {MENU_IMAGESTRING, "selection",      (caddr_t)'s'},
        {MENU_IMAGESTRING, "mod. corps",     (caddr_t)'z'},
                                        };
static struct menu menu_cons_rep_rec_body = {
       MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_rep_rec_item)
       / sizeof(struct menuitem),menu_cons_rep_rec_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_cons_rep_rec_ptr = &menu_cons_rep_rec_body;

/** menu de la gestion des documents en signataire **/
static struct menuitem *menu_ges_sign;
struct menuitem menu_ges_sign_item[] = {
        {MENU_IMAGESTRING, "supprimer",          (caddr_t)'u'},
```

```
        {MENU_IMAGESTRING, "modifier",          (caddr_t)'m'},
        {MENU_IMAGESTRING, "signer",            (caddr_t)'i'},
        {MENU_IMAGESTRING, "mod. note modif.", (caddr_t)'o'},
                                        };
static struct menu menu_ges_sign_body = {
       MENU_IMAGESTRING, "Gestion",sizeof(menu_ges_sign_item)
       / sizeof(struct menuitem),menu_ges_sign_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_ges_sign_ptr = &menu_ges_sign_body;

/** menu des consultations dans le signataire **/
static struct menuitem *menu_cons_sign;
struct menuitem menu_cons_sign_item[] = {
        {MENU_IMAGESTRING, "premier",    (caddr_t)'p'},
        {MENU_IMAGESTRING, "dernier",    (caddr_t)'d'},
        {MENU_IMAGESTRING, "precedent", (caddr_t)'a'},
        {MENU_IMAGESTRING, "suivant",    (caddr_t)'v'},
        {MENU_IMAGESTRING, "selection", (caddr_t)'s'},
                                        };
static struct menu menu_cons_sign_body = {
       MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_sign_item)
       / sizeof(struct menuitem),menu_cons_sign_item,
       &menu_ges_sign_body,(caddr_t)NULL };
static struct menu *menu_cons_sign_ptr = &menu_cons_sign_body;

/** menu de consultation du pere **/
static struct menuitem *menu_cons_pere;
struct menuitem menu_cons_pere_item[] = {
        {MENU_IMAGESTRING, "selection", (caddr_t)'s'},
        {MENU_IMAGESTRING, "mod. corps",(caddr_t)'z'},
                                        };
static struct menu menu_cons_pere_body = {
       MENU_IMAGESTRING, "Consultation",sizeof(menu_cons_pere_item)
       / sizeof(struct menuitem),menu_cons_pere_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_cons_pere_ptr = &menu_cons_pere_body;

/** menu d'administration **/
static struct menuitem *menu_adm;
struct menuitem menu_adm_item[] = {
        {MENU_IMAGESTRING, "gestion journaliere",       (caddr_t)'j'},
        {MENU_IMAGESTRING, "statistiques",              (caddr_t)'s'},
        {MENU_IMAGESTRING, "changement mot de passe",   (caddr_t)'p'},
        {MENU_IMAGESTRING, "changement cle de codage", (caddr_t)'c'},
        {MENU_IMAGESTRING, "creation d'un abonne",      (caddr_t)'f'},
        {MENU_IMAGESTRING, "suppression d'un abonne",   (caddr_t)'d'},
        {MENU_IMAGESTRING, "modification d'un abonne", (caddr_t)'m'},
        {MENU_IMAGESTRING, "description d'un abonne",   (caddr_t)'i'},
                                        };
static struct menu menu_adm_body = {
       MENU_IMAGESTRING, "Administration",sizeof(menu_adm_item)
       / sizeof(struct menuitem),menu_adm_item,
       (struct menu *)NULL,(caddr_t)NULL };
static struct menu *menu_adm_ptr = &menu_adm_body;

/** temoin du choix du menu d'administration **/
static char choix_adm = 'x';

/** tableau contenant la liste ajoutee au bottin **/
static char *lec_fen[] =  {
                        "                                                    ",
```

```
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       "
                                 };

/** tableau utilise pour l'ajout et la suppression **/
/** de nouveaux abonnes                            **/
static char *lec_fen2[] =  {
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       ",
                              "                                       "
                                 };

/** taille des 2 tableaux precedents **/
static int num_lec_fen,num_lec_fen2;

/** compteur de caracteres utilise lors des remplissages des **/
/** tableaux de caracteres                                    **/
static int nbcar;

/** temoin indiquant le fait que le nom de liste a deja ete tape **/
/** ou non                                                        **/
static int abonne;

/** tableau contenant les informations presentes dans la **/
/** subwindow fen2 et leur position                      **/
static struct tab_cont_fen2 {
                      struct rect r;
                      char info_fen2[40];
                         };
static struct tab_cont_fen2 cont_fen2[] =  {
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
                {{0,0,0,0},"                                        "},
```

```
                {{0,0,0,0},"                                                    "},
                {{0,0,0,0},"                                                    "},
                {{0,0,0,0},"                                                    "}
                                                        };

/** taille du tableau cont_fen2 **/
static int lg_cont_fen2;

/** information sur laquelle on a pointe dans fen2 **/
static char info_select[40]="                                                \0";

/** rectangle provisoire **/
static struct rect rerect;

/** string contenant le nom de la liste a supprimer ou un string **/
/** tape dans fen lors d'un transfert                             **/
static char sup[40] = "                                             \0";

/** temoin de creation d'une liste **/
static int newlist = 0;

/** temoin d'edition en cours **/
static int edit_en_cours = 0;

/** temoin de saisie de parametre en cours **/
static int saisie_param = 0;

/** nombre de parametres d'envoi a demander **/
static int rap;

/** nombre de parametres d'envoi deja demandes **/
static int nbmot;

/** temoin du fait que l'utilisateur a deja donne au moins un **/
/** destinataire pour un transfert                            **/
static int au_moins_un = 0;

/** fichier des resumes et des documents **/
FILE *fopen(),*fclose(),*fpres,*fpdoc,*fpdoc_print,*fppere,*fpdoc_mod;
FILE *fpdoc_corps,*fpfils,*fparch,*fpstatp,*fpstatg,*fpdescr;

/** temoin d'occupation de fen **/
static int okfen = 1;

/** temoin de presence des listes sur fen2 **/
static int preslist = 0;

/** temoin de demande du parametre pour le transfert vers **/
/** en attente de traitement                              **/
static int trattrait = 0;

/** numero du document selectionne **/
static int doc_select = 0;

/** tableau des permissions de transfert **/
static int tabperm[10] = {0,0,0,0,0,0,0,0,0,0};

/** indicateurs de presence du document pere **/
static int pere_pres_res = 0;
static int pere_pres_doc = 0;
```

```
/** temoin de consultation du mot de passe **/
static int mp = 0;

/** tableau contenant le mot de passe de l'abonne **/
static char mot_passe[] = "tierce\0                                          ";

/** tableaux contenant les nouveaux mots de passe **/
static char new_mot_passe1[] = "                                             ";
static char new_mot_passe2[] = "                                             ";

/** tableau contenant le mot de passe de confidentialite **/
static char mot_passe_confi[] = "quarte\0                                      ";

/** nombre de mot de passe incorrect deja donne par l'uti. **/
static int nb_mot_passe_false = 0;

/** indice du nombre de mot traites dans la gestion des mots de **/
/** passe                                                       **/
static int newnbmot = 1;

/** temoin d'activation du menu d'attente **/
static int attente = 0;

/** temoin d'edition du rappel **/
static int edit_rappel = 0;

/** compteur du nombre de mots-cles deja tapes **/
static int nb_mot_cles;

/** temoin de la saisie des parametres d'envoi d'un rappel dans fen **/
static int att_rep = 0;

/** temoin d'edition de la note de modification **/
static int edit_note = 0;

/** temoin d'edition de la modification d'un document en signataire **/
static int edit_mod = 0;

/** temoin d'edition de la reponse a un document en attente de **/
/** traitement                                                 **/
static int edit_rep = 0;

/** temoin d'edition du corps d'un document en attente de traitement **/
static int edit_corps = 0;

/** temoin de demande des parametres pour l'archivage **/
static int archive = 0;

/** temoin de consultation d'un seul document des archives **/
static int arch_uniq;

/** temoin de saisie des parametres d'administration dans fen **/
static int adm = 0;

/** mot de passe d'administration **/
static char mot_passe_adm[] = "quintet\0                                       ";

/** cle de codage **/
static char cle_codage[] = "lotto\0                                            ";

/** temoin d'activation du menu d'administration **/
```

```
static int adm_menu = 0;

/*****************************************************************/
/*********************   PROGRAMME PRINCIPAL   *******************/
/*****************************************************************/

main()
{
/** rectangles provisoires **/
struct rect rrr,r1,r2;

/** compteur **/
int i;

/** creation du tool et des subwindows **/
        rect_construct(&rrr,0,0,1152,895);

        if ((tool1 = tool_create("courrier electronique",TOOL_NAMESTRIPE,
                    &rrr,&icon)) == (struct tool*)NULL)
        exit(1);
        font = pw_pfsysopen();

        if ((fen_mes = tool_createsubwindow(tool1,"",
                      TOOL_SWEXTENDTOEDGE,
                      TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
        exit(1);

        if ((fen = tool_createsubwindow(tool1,"",
                   TOOL_SWEXTENDTOEDGE,
                   TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
        exit(1);

        if ((fen2 = tool_createsubwindow(tool1,"",
                    TOOL_SWEXTENDTOEDGE,
                    TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
        exit(1);

        if ((fen3 = tool_createsubwindow(tool1,"",
                    TOOL_SWEXTENDTOEDGE,
                    TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
        exit(1);

        if ((op1 = optsw_createtoolsubwindow(tool1,"",
                   TOOL_SWEXTENDTOEDGE,
                   TOOL_SWEXTENDTOEDGE))  == (struct toolsw *)NULL)
        exit(1);

        if ((op2 = optsw_createtoolsubwindow(tool1,"",
                   TOOL_SWEXTENDTOEDGE,
                   TOOL_SWEXTENDTOEDGE)) == (struct toolsw *)NULL)
        exit(1);

/** initialisation des optionsubwindows **/
        init_option1();
        init_option2();

/** positionnement des subwindows **/
        rect_construct(&rrr,5,23,900,24);
        win_setrect(fen_mes->ts_windowfd,&rrr);

        rect_construct(&rrr,5,206,570,600);
```

```
        win_setrect(fen3->ts_windowfd,&rrr);

        rect_construct(&rrr,5,52,900,149);
        win_setrect(fen->ts_windowfd,&rrr);

        rect_construct(&rrr,5,811,1141,79);
        win_setrect(op1->ts_windowfd,&rrr);

        rect_construct(&rrr,911,23,235,178);
        win_setrect(op2->ts_windowfd,&rrr);

        rect_construct(&rrr,581,206,565,600);
        win_setrect(fen2->ts_windowfd,&rrr);

/** initialisation des windows non-standards **/
        init_fen_mes();
        init_fen();
        init_fen2();
        init_fen3();

/** installation du tool et traitement des signaux **/
        signal(SIGWINCH,sigw);
        tool_install(tool1);
        signal(SIGCHLD,sigc);
        win_setmouseposition(op2->ts_windowfd,40,40);
        tool_select(tool1,50);

/** terminaison du tool (non utilise car sortie par commande) **/
        tool_destroy(tool1);
        exit(0);
}

/*************************************************************************/

/**** gestion des signaux de changement des windows ****/

static sigw() {tool_sigwinch(tool1);}

/*************************************************************************/

/**** gestion du signal de fin de processus ****/

static sigc()
{
                tool_sigchld(tool1);

/** reinitialisation des temoins d'edition **/
/** (edition de nouveau possible)          **/
/** un seul temoin a 1 a la fois           **/
                if (edit_en_cours == 1) edit_en_cours = 0;

                if (edit_note == 1) edit_note = 0;

                if (edit_mod == 1) edit_mod = 0;

                if (edit_corps == 1) edit_corps = 0;

                if (edit_rep == 1) edit_rep = 0;

/** si le temoin d'edition d'un rappel est a 1 ->    **/
/** preparation de fen pour la saisie des parametres **/
```

```
/** du rappel d'un document en attente de reponse     **/
                if (edit_rappel == 1) {
                   att_rep = 1;
                   edit_rappel = 0;
                   pw_writebackground(fen_pixwin,0,0,900,149,PIX_CLR);
        titre("parametres du rappel d'un document en attente de reponse");
                   win_setmouseposition(fen->ts_windowfd,850,50);
                   pw_text(fen_pixwin,20,55,PIX_SRC,NULL,
                           "titre                  : ");
                   pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                           55,PIX_SRC,NULL,'_');
                   pw_text(fen_pixwin,20,75,PIX_SRC,NULL,
                           "confidentiel (o/n) : ");
                   pw_text(fen_pixwin,20,95,PIX_SRC,NULL,
                           "recommande (o/n)   : ");
                   pw_text(fen_pixwin,20,115,PIX_SRC,NULL,
                           "note                   : ");
                   pw_text(fen_pixwin,20,135,PIX_SRC,NULL,
                           "mot-cles               : ");
                   nbcar = 0;
                   nbmot = 1;
                }
}

/*************************************************************************/

/**** fonction d'impression d'un message dans la messagesubw.  ****/

static impmes(text,part)
char text[60];
int part;
{
int i;
                pw_writebackground(fen_mes_pixwin,599,0,2,24,PIX_SET);

/** si le message doit etre imprime dans la partie 1 de la **/
/** sous-fenetre des messages -> clignotement de cette     **/
/** partie et impression de text                           **/
                if (part == 1) {
                   pw_writebackground(fen_mes_pixwin,0,0,598,24,PIX_SET);
                   for (i=0;i<40000;++i);
                   pw_writebackground(fen_mes_pixwin,0,0,598,24,PIX_CLR);
                   pw_text(fen_mes_pixwin,20,16,PIX_SRC,NULL,text);
                }
                else {

/** si le message doit etre imprime dans la partie 2 -> idem dans 2 **/
                pw_writebackground(fen_mes_pixwin,0,0,598,24,PIX_CLR);
                 pw_writebackground(fen_mes_pixwin,601,0,299,24,PIX_SET);
                 for (i=0;i<40000;++i);
                 pw_writebackground(fen_mes_pixwin,601,0,299,24,PIX_CLR);
                 pw_text(fen_mes_pixwin,620,16,PIX_SRC,NULL,text);
                }
}

/*************************************************************************/

/**** fonction d'impression d'un titre dans fen ****/

static titre(text)
char text[100];
```

```
{
                pw_vector(fen_pixwin,0,25,900,25,PIX_SRC,1);
                pw_text(fen_pixwin,20,17,PIX_SRC,NULL,text);
}

/*****************************************************************/
/**********   INITIALISATION DES FENETRES UTILISATEURS   **********/
/*****************************************************************/

/**** initialise fen_mes ****/

static init_fen_mes()
{
struct inputmask mask;

/** cree un pixwin **/
        fen_mes_pixwin = pw_open(fen_mes->ts_windowfd);
        fen_mes_pixwin->pw_prretained = mem_create(900,24,1);

/** definit les procedures a appeler en cas de changement de taille **/
/** ou de positions de la window et en cas d'input dans cette window **/
        fen_mes->ts_io.tio_handlesigwinch = fen_mes_sighandler;
        fen_mes->ts_io.tio_selected = fen_mes_input;

/** initialise le masque des inputs **/
        input_imnull(&mask);

/** met dans le masque les boutons de la souris **/
        win_setinputcodebit(&mask,MS_MIDDLE);
        win_setinputcodebit(&mask,MS_RIGHT);
        win_setinputcodebit(&mask,MS_LEFT);

/** assigne le masque a fen_mes **/
        win_setinputmask(fen_mes->ts_windowfd,&mask,
                         (struct inputmask *)NULL,WIN_NULLLINK);

}

/*****************************************************************/

/**** initialise fen ****/

static init_fen()
{
struct inputmask mask;

/** cree un pixwin **/
        fen_pixwin = pw_open(fen->ts_windowfd);
        fen_pixwin->pw_prretained = mem_create(900,149,1);

/** definit les procedures a appeler en cas de changement de taille **/
/** ou de positions de la window et en cas d'input dans cette window **/
        fen->ts_io.tio_handlesigwinch = fen_sighandler;
        fen->ts_io.tio_selected = fen_input;

/** initialise le masque des inputs **/
        input_imnull(&mask);

/** met dans le masque l'ensemble des caracteres ASCII **/
        mask.im_flags = IM_ASCII;
```

```
/** met dans le masque les boutons de la souris **/
        win_setinputcodebit(&mask,MS_MIDDLE);
        win_setinputcodebit(&mask,MS_RIGHT);
        win_setinputcodebit(&mask,MS_LEFT);

/** assigne le masque a fen **/
        win_setinputmask(fen->ts_windowfd,&mask,
                         (struct inputmask *)NULL,WIN_NULLLINK);

}

/*****************************************************************/

/**** initialise fen2 ****/

static init_fen2()
{
struct inputmask mask;

/** cree un pixwin **/
        fen2_pixwin = pw_open(fen2->ts_windowfd);
        fen2_pixwin->pw_prretained = mem_create(565,600,1);

/** definit les procedures a appeler en cas de changement de taille **/
/** ou de positions de la window et en cas d'input dans cette window **/
        fen2->ts_io.tio_handlesigwinch = fen2_sighandler;
        fen2->ts_io.tio_selected = fen2_input;

/** initialise le masque des inputs **/
        input_imnull(&mask);

/** met dans le masque les boutons de la souris **/
        win_setinputcodebit(&mask,MS_MIDDLE);
        win_setinputcodebit(&mask,MS_RIGHT);
        win_setinputcodebit(&mask,MS_LEFT);

/** assigne le masque a fen2 **/
        win_setinputmask(fen2->ts_windowfd,&mask,
                         (struct inputmask *)NULL,WIN_NULLLINK);
}

/*****************************************************************/

/**** initialise fen3 ****/

static init_fen3()
{
struct inputmask mask;

/** cree un pixwin **/
        fen3_pixwin = pw_open(fen3->ts_windowfd);
        fen3_pixwin->pw_prretained = mem_create(570,600,1);

/** definit les procedures a appeler en cas de changement de taille **/
/** ou de positions de la window et en cas d'input dans cette window **/
        fen3->ts_io.tio_handlesigwinch = fen3_sighandler;
        fen3->ts_io.tio_selected = fen3_input;

/** initialise le masque des inputs **/
        input_imnull(&mask);
```

```
/** met dans le masque les boutons de la souris **/
        win_setinputcodebit(&mask,MS_MIDDLE);
        win_setinputcodebit(&mask,MS_RIGHT);
        win_setinputcodebit(&mask,MS_LEFT);

/** assigne le masque a fen **/
        win_setinputmask(fen3->ts_windowfd,&mask,
                         (struct inputmask *)NULL,WIN_NULLLINK);

}

/*****************************************************************/
/*****   GESTION DES SIGWINCH DANS LES FENETRES UTIL.    ********/
/*****************************************************************/

/**** gere les changements de taille de fen_mes ****/

static fen_mes_sighandler()
{
struct rect rrr;

        rect_construct(&rrr,5,23,900,24);
        win_setrect(fen_mes->ts_windowfd,&rrr);
        pw_damaged(fen_mes_pixwin);
        pw_repairretained(fen_mes_pixwin);
        pw_donedamaged(fen_mes_pixwin);
}

/*****************************************************************/

/**** gere les changements de taille de fen ****/

static fen_sighandler()
{
struct rect rrr;

        rect_construct(&rrr,5,52,900,149);
        win_setrect(fen->ts_windowfd,&rrr);
        pw_damaged(fen_pixwin);
        pw_repairretained(fen_pixwin);
        pw_donedamaged(fen_pixwin);
}

/*****************************************************************/

/**** gere les changements de taille de fen2 ****/

static fen2_sighandler()
{
struct rect rrrr;

        rect_construct(&rrrr,581,206,565,600);
        win_setrect(fen2->ts_windowfd,&rrrr);
        pw_damaged(fen2_pixwin);
        pw_repairretained(fen2_pixwin);
        pw_donedamaged(fen2_pixwin);
}

/*****************************************************************/

/**** gere les changements de taille de fen3 ****/
```

```
static fen3_sighandler()
{
struct rect rrrr;

        rect_construct(&rrrr,5,206,570,600);
        win_setrect(fen3->ts_windowfd,&rrrr);
        pw_damaged(fen3_pixwin);
        pw_repairretained(fen3_pixwin);
        pw_donedamaged(fen3_pixwin);
}

/*****************************************************************/
/******   GESTION DES E/S DANS LES FENETRES UTIL.    ***********/
/*****************************************************************/

/**** initialise cont_fen2 et lg_cont_fen2 ****/

static init_tab_cont_fen2()
{
int i,j;

        for (i=0;i<20;++i)  {
            rect_construct(&rerect,0,0,0,0);
            cont_fen2[i].r = rerect;
                            }
        lg_cont_fen2 = 0;
}

/*****************************************************************/

/**** met le tableau de caractere car dans cont_fen2 a la ****/
/**** position dont le sommet est top, et augmente la     ****/
/**** taille de ce tableau de 1                           ****/

static put_tab_cont_fen2(top,car)
int top;
char car[40];
{
int i;

       rect_construct(&rerect,20,top,
                      strlen(car)*font->pf_defaultsize.x,
                      font->pf_defaultsize.y);
       cont_fen2[lg_cont_fen2].r = rerect;
       for (i=0;i<strlen(car)+1;++i)
           cont_fen2[lg_cont_fen2].info_fen2[i] = car[i];
       ++lg_cont_fen2 ;
}

/*****************************************************************/

/**** prend dans le tableau cont_fen2 l'element dont la partie ****/
/**** rectangle contient le point (x,y) et le met dans         ****/
/**** info_select                                              ****/

static get_tab_cont_fen2(x,y)
int x,y;
{
int i,j,temoin_pres;
struct rect rr;
```

```
        temoin_pres = FALSE;
        i = 0;
        while ((temoin_pres == FALSE) && (i<lg_cont_fen2)) {
              rr = cont_fen2[i].r;
              temoin_pres = rect_includespoint(&rr,x,y);
              ++i;
              }
        if (temoin_pres == TRUE)
           for (j=0;j<strlen(cont_fen2[i-1].info_fen2)+1;++j)
               info_select[j] = cont_fen2[i-1].info_fen2[j];
}

/*****************************************************************/

/**** imprime dans fen2 la liste dont le nom est contenu dans ****/
/**** info_select et remplit cont_fen2 parallellement.        ****/
/**** reinitialise info_select                                ****/

static imprim_liste()
{
int i,j;
char *temp,*temp2;


/** si le contenu d'info_select est egal a "liste des abbes" ->    **/
/** initialisation de cont_fen2, effacement de fen2, et impression **/
/** dans fen2 de la liste des abbes                                **/
        if (strcmp(info_select,"liste des abbes\0") == 0) {
           init_tab_cont_fen2();
           pw_writebackground(fen2_pixwin,0,0,567,600,PIX_CLR);
           pw_text(fen2_pixwin,20,30,PIX_SRC,NULL,info_select);
           put_tab_cont_fen2(14,info_select);
           pw_text(fen2_pixwin,20,45,PIX_SRC,NULL,"---------------");
           for (i=0;i<lg_tababbe;++i) {
             pw_text(fen2_pixwin,20,30*(i+3),PIX_SRC,NULL,tababbe[i]);
             put_tab_cont_fen2(30*(i+3) - font->pf_defaultsize.y,
                               tababbe[i]);
           }
        }
/** idem avec liste des profs **/
       if (strcmp(info_select,"liste des profs\0") == 0) {
           init_tab_cont_fen2();
           pw_writebackground(fen2_pixwin,0,0,567,600,PIX_CLR);
           pw_text(fen2_pixwin,20,30,PIX_SRC,NULL,info_select);
           put_tab_cont_fen2(14,info_select);
           pw_text(fen2_pixwin,20,45,PIX_SRC,NULL,"---------------");
       for (i=0;i<lg_tabprof;++i) {
              pw_text(fen2_pixwin,20,30*(i+3),PIX_SRC,NULL,tabprof[i]);
              put_tab_cont_fen2(30*(i+3) - font->pf_defaultsize.y,
                               tabprof[i]);
          }
        }

/** idem avec liste des eleves **/
        if (strcmp(info_select,"liste des eleves\0") == 0) {
           init_tab_cont_fen2();
           pw_writebackground(fen2_pixwin,0,0,567,600,PIX_CLR);
           pw_text(fen2_pixwin,20,30,PIX_SRC,NULL,info_select);
           put_tab_cont_fen2(14,info_select);
           pw_text(fen2_pixwin,20,45,PIX_SRC,NULL,"---------------");
```

```
            for (i=0;i<lg_tabelev;++i) {
                pw_text(fen2_pixwin,20,30*(i+3),PIX_SRC,NULL,tabelev[i]);
                put_tab_cont_fen2(30*(i+3) - font->pf_defaultsize.y,
                                  tabelev[i]);
            }
        }

/** idem avec liste des abonnes **/
        if (strcmp(info_select,"liste des abonnes\0") == 0) {
            init_tab_cont_fen2();
            pw_writebackground(fen2_pixwin,0,0,567,600,PIX_CLR);
            pw_text(fen2_pixwin,20,30,PIX_SRC,NULL,info_select);
            put_tab_cont_fen2(14,info_select);
            pw_text(fen2_pixwin,20,45,PIX_SRC,NULL,"-----------------");
            for (i=0;i<lg_taball;++i) {
                pw_text(fen2_pixwin,20,30*(i+3),PIX_SRC,NULL,taball[i]);
                put_tab_cont_fen2(30*(i+3) - font->pf_defaultsize.y,
                                  taball[i]);
        }
        }

/** idem avec lec_fen (qui contient le nom de la liste que **/
/** l'utilisateur a ajoute)                                **/
        temp2 = "----------------------------------------\0";
        if (strcmp(info_select,*lec_fen) == 0) {
            if (newlist == 1)  {
                init_tab_cont_fen2();
                pw_writebackground(fen2_pixwin,0,0,567,600,PIX_CLR);
                pw_text(fen2_pixwin,20,30,PIX_SRC,NULL,info_select);
                put_tab_cont_fen2(14,info_select);
                for (i=0;i<strlen(*lec_fen);++i) temp2[i] = '-';
                temp2[i] = '\0';
                pw_text(fen2_pixwin,20,45,PIX_SRC,NULL,temp2);
                for (i=0;i<num_lec_fen - 1;++i) {
                    pw_text(fen2_pixwin,20,30*(i+3),
                            PIX_SRC,NULL,lec_fen[i+1]);
                    put_tab_cont_fen2(30*(i+3) - font->pf_defaultsize.y,
                                      lec_fen[i + 1]);
                }
            }
        }

/** reinitialisation d'info_select **/
        strcpy(info_select,"\0");
}

/*************************************************************/

/**** gere le menu de la consultation du bottin ****/

static ges_menu1_fen2()
{
char *text="nom de la liste : \0";
int i;

/** impression du menu et saisie du choix de l'utilisateur **/
        if (ievent_fen2.ie_code ==
            MENU_BUT && win_inputposevent(&ievent_fen2)
            && (menu1_fen2 = menu_display(&menu1_fen2_ptr,&ievent_fen2,
            fen2->ts_windowfd)))
            choice_menu1_fen2 = (char) menu1_fen2->mi_data;
```

```
        if (okfen == 1) {
         switch (choice_menu1_fen2) {

/** cas de la creation d'une liste **/
                case 'c' : impmes("  CREATION D'UNE LISTE",1);
                           okfen = 0;

/** reinitialisation de l'element de tabbot qui va contenir le nom **/
/** de la nouvelle liste dans le cas ou une liste a deja ete creee **/
                           if (newlist == 1) tabbot[lg_tabbot] = "\0";

/** preparation de fen pour la demande de la liste a creer **/
/** et de ses elements                                     **/
                           win_setmouseposition(fen->ts_windowfd,850,50);
                           pw_writebackground(fen_pixwin,0,0,
                                              900,149,PIX_CLR);
                           titre("creation d'une liste");
                           pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text);
                           pw_char(fen_pixwin,
                                   50 + 16*font->pf_defaultsize.x,
                                   55,PIX_SRC,NULL,'_');
                           nbcar = 0;
                           abonne = 0;
                           num_lec_fen = 0;
                           break;

/** cas de la suppression d'une liste **/
                case 's' : impmes("  SUPPRESSION D'UNE LISTE",1);
                           okfen = 0;

/** preparation de fen pour la demande du nom de la liste **/
/** a supprimer                                           **/
                           win_setmouseposition(fen->ts_windowfd,850,50);
                           pw_writebackground(fen_pixwin,0,0,
                                              900,149,PIX_CLR);
                           titre("suppression d'une liste");
                           pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text);
                           pw_char(fen_pixwin,
                                   50 + 16*font->pf_defaultsize.x,
                                   55,PIX_SRC,NULL,'_');
                           nbcar = 0;
                           break;

/** cas de l'ajout d'un ou plusieurs abonnes a une liste **/
                case 'a' : impmes("  AJOUT D'UN ELEMENT",1);
                           okfen = 0;

/** preparation de fen pour la demande de la liste dans laquelle on **/
/** veut rajouter des elements et de ces elements                   **/
                           win_setmouseposition(fen->ts_windowfd,850,50);
                           pw_writebackground(fen_pixwin,0,0,
                                              900,149,PIX_CLR);
                           titre("ajout d'abonnes dans une liste");
                           pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text);
                           pw_char(fen_pixwin,
                                   50 + 16*font->pf_defaultsize.x,
                                   55,PIX_SRC,NULL,'_');
                           nbcar = 0;
                           abonne = 0;
                           num_lec_fen2 = 0;
```

```
                        break;

/** cas de la suppression d'un ou plusieurs elements d'une liste **/
              case 'd' : impmes("  SUPPRESSION D'UN ELEMENT",1);
                        okfen = 0;

/** preparation de fen pour la demande de la liste dans laquelle on **/
/** veut supprimer des elements et de ces elements                  **/
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_writebackground(fen_pixwin,0,0,
                                           900,149,PIX_CLR);
                        titre("suppression d'abonnes dans une liste");
                        pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text);
                        pw_char(fen_pixwin,
                                50 + 16*font->pf_defaultsize.x,
                                55,PIX_SRC,NULL,'_');
                        nbcar = 0;
                        abonne = 0;
                        num_lec_fen2 = 0;
                        break;

                      }

          }
          else {
               impmes("  INTERDIT POUR LE MOMENT",1);
               choice_menu1_fen2 = 'x';
          }
}

/*****************************************************************/

/**** fonction s'occupant de la terminaison des commandes ****/
/**** offertes par le menu de consultation du bottin      ****/

static fin_fen_input()
{
int i,j,k,l,num,chnew;

     switch(choice_menu1_fen2) {

/** creation d'une liste **/
       case 'c' :
                 pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,85,
                         PIX_SRC,NULL,' ');
                 if (num_lec_fen == 0) {

/** l'utilisateur n'a pas tape de nom de liste **/
                    impmes("  CREATION AVORTEE",1);
                 }
                 else {

/** l'utilisateur a tape un nom de liste -> recopiage dans la **/
/** liste des listes                                          **/
                 strcpy(*(tabbot+lg_tabbot),*lec_fen);
                 impmes("  CREATION EFFECTUEE",1);
                 newlist = 1;
                 }

/** reinitialisation du choix du menu de consultation du bottin **/
                 choice_menu1_fen2 = 'x';
```

```
                    break;

/** suppression d'une liste **/
        case 's' :
                    pw_char(fen_pixwin,50+(16+nbcar)*font->pf_defaultsize.x,
                            55,PIX_SRC,NULL,' ');

/** reinitialisation du choix du menu de consultation du bottin **/
                    choice_menu1_fen2 = 'x';
                    if (nbcar == 0) {

/** l'utilisateur n'a pas donne de nom de liste **/
                       impmes("  SUPPRESSION AVORTEE",1);
                    }
                    else {

/** l'utilisateur a tape un nom de liste **/
                     if (strcmp(sup,"liste des abonnes\0")==0) {

/** le nom de liste tape est la liste des abonnes, qu'on ne peut **/
/** pas supprimer                                                **/
            impmes("  INTERDICTION DE SUPPRIMER LA LISTE DES ABONNES",1);
                     }
                     else {

/** le nom donne est compare aux noms existants **/
                       if (exist_liste(sup,&num) == 0) {

/** le nom n'existe pas **/
                         impmes("  CETTE LISTE N'EXISTE PAS",1);
                       }
                       else {

/** le nom existe et on le supprime de la liste des listes **/
                         impmes("  LISTE SUPPRIMEE",1);
                         chnew=0;

/** si la liste supprimee est celle qui a deja ete creee -> remise a **/
/** zero de newlist                                                  **/
                         if ((strcmp(sup,tabbot[lg_tabbot-1+newlist])==0)
                              && (newlist==1))
                            chnew=1;
                         for (i=num;i<lg_tabbot+newlist;++i)
                           strcpy(*(tabbot+i),*(tabbot+i+1));
                         if (chnew==1)
                              newlist = 0;
                         else lg_tabbot-=1;
                       }
                     }
                   }
                   break;


/** ajout d'elements a une liste **/
        case 'a' :
                    pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,85,
                            PIX_SRC,NULL,' ');

/** reinitialisation du choix du menu de consultation du bottin **/
                    choice_menu1_fen2 = 'x';
                    if (num_lec_fen2 == 0) {
```

```
/** le nom de liste donne est vide **/
                    impmes("  AJOUT AVORTE",1);
                    }
                 else {

/** le nom de liste donne est compare aux noms de listes existants **/
/** et les abonnes donnes sont ajoutes a cette liste               **/
                 if (strcmp(*lec_fen2,"liste des abbes\0")==0) {
                    for (i=1;i<num_lec_fen2;++i)
                        strcpy(*(tababbe + lg_tababbe + i - 1),
                               *(lec_fen2 + i));
                    lg_tababbe+=num_lec_fen2-1;
                    }
                 if (strcmp(*lec_fen2,"liste des profs\0")==0) {
                    for (i=1;i<num_lec_fen2;++i)
                        strcpy(*(tabprof + lg_tabprof + i - 1),
                               *(lec_fen2 + i));
                    lg_tabprof+=num_lec_fen2-1;
                    }
                 if (strcmp(*lec_fen2,"liste des eleves\0")==0) {
                    for (i=1;i<num_lec_fen2;++i)
                        strcpy(*(tabelev + lg_tabelev + i - 1),
                               *(lec_fen2 + i));
                    lg_tabelev+=num_lec_fen2-1;
                    }
                 if (strcmp(*lec_fen2,*lec_fen)==0) {
                    for (i=1;i<num_lec_fen2;++i)
                        strcpy(*(lec_fen + num_lec_fen - 1 + i ),
                               *(lec_fen2 + i));
                    num_lec_fen+=num_lec_fen2-1;
                    }
                 impmes("  AJOUT EFFECTUE",1);
                 }
                 break;


/** suppression d'elements a une liste **/
        case 'd' :
                  pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,85,
                          PIX_SRC,NULL,' ');

/** reinitialisation du choix du menu de consultation du bottin **/
                 choice_menu1_fen2 = 'x';
                 if (num_lec_fen2 == 0) {

/** le nom de liste donne est vide **/
                    impmes("  DELETE AVORTE",1);
                    }
                 else {

/** le nom de liste donne est compare aux noms de listes existants **/
/** et les abonnes donnes sont supprime a cette liste              **/
                 if (strcmp(*lec_fen2,"liste des abbes\0")==0)
                    for (i=1;i<num_lec_fen2;++i) {
                     for (j=0;strcmp(tababbe[j],lec_fen2[i]) != 0;++j);
                     for (k=j;k<lg_tababbe;++k)
                      strcpy(tababbe[k],tababbe[k+1]);
                     lg_tababbe-=1;
                    }
                 if (strcmp(*lec_fen2,"liste des profs\0")==0)
```

```
                    for (i=1;i<num_lec_fen2;++i) {
                     for (j=0;strcmp(tabprof[j],lec_fen2[i]) != 0;++j);
                     for (k=j;k<lg_tabprof;++k)
                      strcpy(tabprof[k],tabprof[k+1]);
                     lg_tabprof-=1;
                    }
                  if (strcmp(*lec_fen2,"liste des eleves\0")==0)
                    for (i=1;i<num_lec_fen2;++i) {
                     for (j=0;strcmp(tabelev[j],lec_fen2[i]) != 0;++j);
                     for (k=j;k<lg_tabelev;++k)
                      strcpy(tabelev[k],tabelev[k+1]);
                     lg_tabelev-=1;
                    }
                  if (strcmp(*lec_fen2,*lec_fen)==0)
                    for (i=1;i<num_lec_fen2;++i) {
                     for (j=0;strcmp(lec_fen[j+1],lec_fen2[i])!=0;++j);
                     for (k=j;k<num_lec_fen;++k)
                      strcpy(lec_fen[k+1],lec_fen[k+2]);
                     num_lec_fen-=1;
                    }
                impmes("  DELETE EFFECTUE",1);
                }
                break;
        }
        okfen = 1;
}

/*****************************************************************/

/**** imprime dans fen le caractere caract a la position (x,y) ****/
/**** renvoie 1 dans deletion si le caractere tape est un del  ****/
/**** note : maximum 30 caracteres par nom                     ****/

static imprim_char_suiv_fen(x,y,caract,deletion,nb)
int x,y,*deletion,nb;
char caract;
{
       if (nb < 30) {
          *deletion = 0;
          if ((caract == 127)  && (nb > 0)) {

/** le caractere est un del et l'utilisateur a deja tape un autre    **/
/** caractere -> on efface le caractere precedent et on fait reculer **/
/** le curseur                                                       **/
             pw_char(fen_pixwin,x - font->pf_defaultsize.x,y+25,
                     PIX_SRC,NULL,'_');
             *deletion = 1;
             pw_char(fen_pixwin,x,y+25,PIX_SRC,NULL,' ');
          }
          else  if (caract != 127 ) {

/** le caractere n'est pas un del -> on l'imprime et on fait avancer **/
/** le curseur ou, dans le cas d'un mot de passe, on imprime ~       **/
                if ((mp == 1) || (((nbmot == 1) || (nbmot == 9)
                    || (nbmot == 10) || (nbmot == 11)
                    || (nbmot == 12) || (nbmot == 13)) && (adm == 1)))
                   pw_char(fen_pixwin,x,y+25,PIX_SRC,NULL,'~');
                else pw_char(fen_pixwin,x,y+25,PIX_SRC,NULL,caract);
                pw_char(fen_pixwin,x + font->pf_defaultsize.x,
                        y+25,PIX_SRC,NULL,'_');
                }
```

```
            }
        else {

/** plus de 30 caracteres tapes -> refuse sauf si c'est un del **/
          impmes("  PAS PLUS DE 30 CARACTERES",1);
          if (caract == 127) {
             pw_char(fen_pixwin,x - font->pf_defaultsize.x,y+25,
                     PIX_SRC,NULL,'_');
             *deletion = 1;
             pw_char(fen_pixwin,x,y+25,PIX_SRC,NULL,' ');
          }
         }
}

/*****************************************************************/

/**** renvoie 1 si le nom de l'abonne chaine est deja present ****/
/**** dans la liste chaine2                                   ****/

static exist_abonne(chaine,chaine2)
char chaine[],chaine2[];
{
int i,trouve;

        trouve = 0;
        i = 0;
        if (strcmp(chaine2,"liste des abbes\0")==0)
           while ((trouve == 0) && (i<lg_tababbe)) {
              if (strcmp(chaine,tababbe[i]) == 0) trouve = 1;
              ++i;
           }
        if (strcmp(chaine2,"liste des profs\0")==0)
           while ((trouve == 0) && (i<lg_tabprof)) {
              if (strcmp(chaine,tabprof[i]) == 0) trouve = 1;
              ++i;
           }
        if (strcmp(chaine2,"liste des eleves\0")==0)
           while ((trouve == 0) && (i<lg_tabelev)) {
              if (strcmp(chaine,tabelev[i]) == 0) trouve = 1;
              ++i;
           }
        if (strcmp(chaine2, *lec_fen)==0)
           while ((trouve == 0) && (i<num_lec_fen)) {
              if (strcmp(chaine,lec_fen[i]) == 0) trouve = 1;
              ++i;
           }
        if (strcmp(chaine2,"liste des abonnes\0")==0)
           while ((trouve == 0) && (i<lg_taball)) {
              if (strcmp(chaine,taball[i]) == 0) trouve = 1;
              ++i;
           }
        return(trouve);
}

/*****************************************************************/

/**** renvoie 1 si le nom de la liste chaine est deja present    ****/
/**** dans la liste des listes . i est l'indice de cette liste   ****/
/**** dans la liste des listes                                   ****/

static exist_liste(chaine,i)
```

```
char chaine[];
int *i;
{
int trouve;

        trouve = 0;
        *i = 0;
        while ((trouve == 0) && (*i<lg_tabbot+newlist)) {
              if (strcmp(chaine,tabbot[*i]) == 0) trouve = 1;
              ++(*i);
        }
        --(*i);
        return(trouve);
}

/*****************************************************************/

/**** cette fonction est appelee asynchronement chaque fois ****/
/**** qu'un caractere est tape dans fen .                   ****/
/**** elle s'occupe d'imprimer le caractere tape par        ****/
/**** l'utilisateur dans fen a la bonne position (avec      ****/
/**** gestion de la touche del) et de traiter ce caractere. ****/
/**** de plus, elle verifie la validite des informations    ****/
/**** donnees                                               ****/

static fen_input(sw,ibits,ebits,obits,timer)
caddr_t sw;
int *ibits,*ebits,*obits;
struct timeval **timer;

{
int coordx,coordy,efface;

/** lecture de l'evenement survenu dans fen (seuls les caracteres **/
/** et les boutons de la souris sont acceptes comme evenement     **/
/** dans le masque input de fen)                                  **/
        input_readevent(fen->ts_windowfd,&ievent_fen);
        *ibits = *obits = *ebits = 0;

/** si l'utilisateur a appuye sur un des boutons de la souris -> **/
/** on ne fait rien                                               **/
        if ((ievent_fen.ie_code == MS_LEFT) ||
            (ievent_fen.ie_code == MS_RIGHT) ||
            (ievent_fen.ie_code == MS_MIDDLE)) ;
        else {

/** si l'utilisateur tape la touche ESC -> reinitialisation des **/
/** temoins d'utilisation de fen, effacage de fen et message de **/
/** confirmation de l'avortement                                **/
           if (ievent_fen.ie_code == 27) {
              saisie_param = 0;
              mp = 0;
              att_rep = 0;
              archive = 0;
              adm = 0;
              choice_menu1_fen2 = 'x';
              okfen = 1;
              pw_writebackground(fen_pixwin,0,0,900,149,PIX_CLR);
              impmes("  OPERATION AVORTEE",1);
           }
           else {
```

```
/** le caractere est traite selon le choix que l'utilisateur a fait **/
/** dans le menu de la consultation du bottin                       **/
          switch (choice_menu1_fen2) {

             case 'c' : creer_liste();
                        break;

             case 's' : supprimer_liste();
                        break;

             case 'a' : ajout_abonne();
                        break;

             case 'd' : suppress_abonne();
                        break;

           }

/** selon le temoin d'utilisation de fen -> traitement des **/
/** caracteres tapes dans fen                              **/
           if (saisie_param == 1) trait_saisie_param();
           if (mp == 1) trait_mot_passe();
           if (att_rep == 1) trait_att_rep();
           if (archive == 1) trait_archive();
           if (adm == 1) trait_adm();
        }
     }
}

/*****************************************************************/

/**** lecture des parametres d'envoi d'un document ****/

static trait_saisie_param()
{
int coordx,coordy,efface;

            if (ievent_fen.ie_code == 13)

/** si l'utilisateur tape return -> gestion du parametre que **/
/** l'utilisateur vient de terminer                          **/
                switch (nbmot) {

                      case 1 : saisie_1();
                               break;

                      case 2 : saisie_2_3();
                               break;

                      case 3 : saisie_2_3();
                               break;

                      case 4 : saisie_4();
                               break;

                      case 5 : saisie_5();
                               break;

                      case 6 : saisie_6();
                               break;
```

```
                        case 7 : saisie_7();
                                 break;

                        case 8 : saisie_8();
                                 break;

                        case 9 : saisie_9();
                                 break;

                        case 10 : saisie_10();
                                  break;

                        case 11 : saisie_11();
                                  break;

                        case 12 : saisie_12();
                                  break;
                    }
                else {

/** si l'utilisateur tape un caractere ou del -> impression de ce **/
/** caractere dans fen et incrementation du compteur de caractere **/
                    coordx = 30 + (21+nbcar)*font->pf_defaultsize.x;
                    coordy = nbmot*20;
                    if (nbmot > 6 )  {
                       coordx +=450;
                       coordy = 20*(nbmot-6);
                    }
                    imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                         &efface,nbcar);
                    if (efface == 1) nbcar--;
                    else
                       if (nbcar<30) {
                          *(sup + nbcar) = ievent_fen.ie_code;
                          nbcar++;
                       }
                }
}

/*****************************************************************/

/**** lecture des parametres pour l'administration ****/

static trait_adm()
{
int efface,coordx,coordy;


/** si le caractere tape dans fen est RETURN (code ascii 13) ->  **/
/** gestion du parametre que l'utilisateur vient de finir de     **/
/** taper et preparation de fen pour le parametre suivant a lire **/
/** nbmot est le numero du parametre que l'utilisateur est en    **/
/** train de taper                                               **/
      if (ievent_fen.ie_code == 13) {

         switch (nbmot) {

            case 1 : *(sup + nbcar) = '\0';
```

```
/** si le parametre tape est egal au mot de passe d'administration -> **/
                    if (strcmp(sup,mot_passe_adm) == 0) {
                       adm_menu = 1;
                       pw_char(fen_pixwin,
                               20+(32+nbcar)*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,' ');
                       pw_writebackground(fen2_pixwin,0,0,565,600,
                                          PIX_CLR);
                       pw_writebackground(fen3_pixwin,0,0,570,600,
                                          PIX_CLR);
                       pw_text(fen3_pixwin,20,20,PIX_SRC,NULL,
                     "veuillez activer le menu avec le bouton de droite");
                       adm = 0;
                       okfen = 1;
                    }

/** sinon message d'erreur et liberation de fen **/
                    else {
                       pw_writebackground(fen_pixwin,0,0,900,149,
                                          PIX_CLR);
                       impmes("  MOT DE PASSE ERRONNE",1);
                       adm = 0;
                       okfen = 1;
                    }
                    break;

            case 2 : adm = 0;
                    pw_char(fen_pixwin,
                            20+(16+nbcar)*font->pf_defaultsize.x,
                            65,PIX_SRC,NULL,' ');
                    okfen = 1;
                    nbcar = 0;
                    break;

            case 3 : pw_char(fen_pixwin,
                            20+(32+nbcar)*font->pf_defaultsize.x,
                            45,PIX_SRC,NULL,' ');
                    *(sup + nbcar) = '\0';

/** si le parametre lu est egal a 'a' -> impression du mot **/
/** 'abonne' a la place du 'a' et affichage des libelles   **/
/** pour les parametres suivants a demander                **/
                    if (strcmp(sup,"a\0") == 0) {
                       pw_text(fen_pixwin,20+32*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"abonne");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                               "identifiant d'abonne          : _");
                       pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                               "date de debut                 :  ");
                       pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                               "date de fin                   :  ");
                       nbmot = 4;
                    }
                    else

/** idem avec 'r' et 'rapport general' **/
                    if (strcmp(sup,"r\0") == 0) {
                       pw_text(fen_pixwin,20+32*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"rapport general");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                               "date de debut                 : _");
```

```
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                                "date de fin                    :  ");
                        nbmot = 7;
                    }
                    else {

/** si l'utilisateur n'a tape ni 'a' ni 'r' -> on efface ce **/
/** qu'il a tape et on attend qu'il retape ce parametre     **/
                        pw_writebackground(fen_pixwin,
                                           20+32*font->pf_defaultsize.x,
                                           29,500,20,PIX_CLR);
                        pw_char(fen_pixwin,20+32*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,'_');
                    }
                    nbcar = 0;
                    break;

            case 4 : pw_char(fen_pixwin,
                             20+(32+nbcar)*font->pf_defaultsize.x,
                             65,PIX_SRC,NULL,' ');
                     pw_char(fen_pixwin,20+32*font->pf_defaultsize.x,
                             85,PIX_SRC,NULL,'_');
                     nbmot = 5;
                     nbcar = 0;
                     break;

            case 5 : pw_char(fen_pixwin,
                             20+(32+nbcar)*font->pf_defaultsize.x,
                             85,PIX_SRC,NULL,' ');
                     pw_char(fen_pixwin,20+32*font->pf_defaultsize.x,
                             105,PIX_SRC,NULL,'_');
                     nbmot = 6;
                     nbcar = 0;
                     break;

            case 6 : pw_char(fen_pixwin,
                             20+(32+nbcar)*font->pf_defaultsize.x,
                             105,PIX_SRC,NULL,' ');
                     nbcar = 0;

/** impression dans fen2 des statistiques particulieres **/
                     pw_writebackground(fen2_pixwin,0,0,565,600,PIX_CLR);
                     fpstatp = fopen("statp","r");
                     imprime(fpstatp,1,2);
                     fclose(fpstatp);
                     adm = 0;
                     okfen = 1;
                     break;

            case 7 : pw_char(fen_pixwin,
                             20+(32+nbcar)*font->pf_defaultsize.x,
                             65,PIX_SRC,NULL,' ');
                     pw_char(fen_pixwin,20+32*font->pf_defaultsize.x,
                             85,PIX_SRC,NULL,'_');
                     nbmot = 8;
                     nbcar = 0;
                     break;

            case 8 : pw_char(fen_pixwin,
                             20+(32+nbcar)*font->pf_defaultsize.x,
                             85,PIX_SRC,NULL,' ');
```

```
/** impression dans fen2 des statistiques generales **/
                    pw_writebackground(fen2_pixwin,0,0,565,600,PIX_CLR);
                    fpstatg = fopen("statg","r");
                    imprime(fpstatg,1,2);
                    fclose(fpstatg);
                    adm = 0;
                    okfen = 1;
                    nbcar = 0;
                    break;

            case 9 : *(sup + nbcar) = '\0';
                    strcpy(new_mot_passe1,sup);
                    pw_char(fen_pixwin,
                            20+(23+nbcar)*font->pf_defaultsize.x,
                            45,PIX_SRC,NULL,' ');
                    pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                            "nouveau mot de passe : _");
                    nbcar = 0;
                    nbmot = 10;
                    break;

          case 10 : *(sup + nbcar) = '\0';

/** si le parametre qui vient d'etre tape est egal au parametre **/
/** precedent -> modification du mot de passe d'administration  **/
                    if (strcmp(sup,new_mot_passe1) == 0) {
                       strcpy(mot_passe_adm,sup);
                       impmes("  MOT DE PASSE MODIFIE",1);
                    }
                    else impmes("  MOT DE PASSE INCHANGÉ",1);
                    pw_char(fen_pixwin,
                            20+(23+nbcar)*font->pf_defaultsize.x,
                            65,PIX_SRC,NULL,' ');
                    okfen = 1;
                    adm = 0;
                    nbcar = 0;
                    break;

          case 11 : *(sup + nbcar) = '\0';

/** si le parametre tape est egal a la cle de codage -> impression **/
/** du libelle pour la nouvelle cle de codage                      **/
                    if (strcmp(sup,cle_codage) == 0) {
                       pw_char(fen_pixwin,
                               20+(25+nbcar)*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,' ');
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                               "nouvelle cle de codage : _");
                       nbcar = 0;
                       nbmot = 12;
                    }
                    else {

/** sinon message d'erreur **/
                       impmes("  CLE DE CODAGE INCORRECTE",1);
                       pw_writebackground(fen_pixwin,0,0,900,149,
                                          PIX_CLR);
                       okfen = 1;
                       adm = 0;
                    }
```

```
                    break;

          case 12 : *(sup + nbcar) = '\0';

/** copie de la nouvelle cle de codage et redemande la nouvelle **/
/** cle de codage pour confirmation                             **/
                    strcpy(new_mot_passe1,sup);
                    pw_char(fen_pixwin,
                            20+(25+nbcar)*font->pf_defaultsize.x,
                            65,PIX_SRC,NULL,' ');
                    pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                            "nouvelle cle de codage : _");
                    nbcar = 0;
                    nbmot = 13;
                    break;

          case 13 : *(sup + nbcar) = '\0';

/** si les deux nouvelles cles de codage sont egales -> modification **/
/** de la cle de codage et message de confirmation                   **/
                    if (strcmp(sup,new_mot_passe1) == 0) {
                       strcpy(cle_codage,sup);
                       impmes("  CLE DE CODAGE MODIFIEE",1);
                    }

/** sinon pas de modification **/
                    else impmes("  CLE DE CODAGE INCHANGEE",1);
                    pw_char(fen_pixwin,
                            20+(25+nbcar)*font->pf_defaultsize.x,
                            85,PIX_SRC,NULL,' ');
                    okfen = 1;
                    adm = 0;
                    nbcar = 0;
                    break;

          case 14 : pw_char(fen_pixwin,
                            20+(53+nbcar)*font->pf_defaultsize.x,
                            45,PIX_SRC,NULL,' ');
                    *(sup + nbcar) = '\0';

/** si le parametre tape est egal a 's' -> impression de 'service' **/
/** a la place de 's' et impression des libelles des parametres    **/
/** suivants                                                       **/
                    if (strcmp(sup,"s\0") == 0) {
                       pw_text(fen_pixwin,20+53*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"service");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
              "identifiant du service                                : _");
                       pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
              "identifiant du patron du service                      :  ");
                    nbmot = 15;
                    }
                    else

/** si le parametre est egal a 'p' -> impression de 'patron' a la   **/
/** place de 'p' et impression des libelles des parametres suivants **/
                    if (strcmp(sup,"p\0") == 0) {
                       pw_text(fen_pixwin,20+53*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"patron");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
              "identifiant du patron                                 : _");
```

```
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                "identifiant du service de ce patron                    :  ");
                        nbmot = 15;
                        }
                        else

/** si le parametre est egal a 'd' -> impression de 'dactylo' a la  **/
/** place de 'd' et impression des libelles des parametres suivants **/
                        if (strcmp(sup,"d\0") == 0) {
                           pw_text(fen_pixwin,20+53*font->pf_defaultsize.x,
                                   45,PIX_SRC,NULL,"dactylo");
                           pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                "identifiant du service                                 : _");
                           pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                "identifiant du service                                 : ");
                        nbmot = 15;
                        }
                        else

/** si le parametre est egal a 'se' -> impression de 'secretaire' a     **/
/** la place de 'se' et impression des libelles des parametres suivants **/
                        if (strcmp(sup,"se\0") == 0) {
                           pw_text(fen_pixwin,20+53*font->pf_defaultsize.x,
                                   45,PIX_SRC,NULL,"secretaire");
                           pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                "identifiant de la secretaire                           : _");
                           pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                "identifiant du service                                 : ");
                        nbmot = 15;
                        }
                        else {

/** si le parametre n'est egal ni a 's',ni a 'p', ni a 'd', **/
/** ni a 'se' -> on efface ce que l'utilisateur a tape et   **/
/** on attend qu'il retape ce parametre                     **/
                           pw_writebackground(fen_pixwin,
                                              20+53*font->pf_defaultsize.x,
                                              29,500,20,PIX_CLR);
                           pw_char(fen_pixwin,20+53*font->pf_defaultsize.x,
                                   45,PIX_SRC,NULL,'_');
                        }
                        nbcar = 0;
                        break;

                case 15 : pw_char(fen_pixwin,20+53*font->pf_defaultsize.x,
                                  85,PIX_SRC,NULL,'_');
                        pw_char(fen_pixwin,
                                  20+(53+nbcar)*font->pf_defaultsize.x,
                                  65,PIX_SRC,NULL,' ');
                        nbcar = 0;
                        nbmot = 16;
                        break;

                case 16 : pw_char(fen_pixwin,
                                  20+(53+nbcar)*font->pf_defaultsize.x,
                                  85,PIX_SRC,NULL,' ');
                        adm = 0;
                        okfen = 1;
                        nbcar = 0;
                        break;
```

```
            case 17 : pw_char(fen_pixwin,
                              20+(21+nbcar)*font->pf_defaultsize.x,
                              45,PIX_SRC,NULL,' ');
                      pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                              "confirmation (o/n) : _");
                      nbcar = 0;
                      nbmot = 18;
                      break;

            case 18 : *(sup + nbcar) = '\0';

/** si le parametre tape est egal a 'o' -> message de confirmation **/
/** de la suppression                                              **/
                      if (strcmp(sup,"o\0") == 0) {
                         pw_char(fen_pixwin,
                                 20+(21+nbcar)*font->pf_defaultsize.x,
                                 65,PIX_SRC,NULL,' ');
                         adm = 0;
                         okfen = 1;
                         impmes("  SUPPRESSION EFFECTUEE",1);
                      }
                      else

/** si le parametre tape est egal a 'n' -> message d'infirmation **/
/** de la suppression                                           **/
                      if (strcmp(sup,"n\0") == 0) {
                         pw_char(fen_pixwin,
                                 20+(21+nbcar)*font->pf_defaultsize.x,
                                 65,PIX_SRC,NULL,' ');
                         adm = 0;
                         okfen = 1;
                         impmes("  SUPPRESSION AVORTEE",1);
                      }
                      else {

/** si le parametre n'est egal ni a 'o', ni a 'n' -> on efface ce **/
/** qu'il a tape et on attend qu'il retape ce parametre           **/
                         pw_writebackground(fen_pixwin,
                                            20+21*font->pf_defaultsize.x,
                                            49,500,20,PIX_CLR);
                         pw_char(fen_pixwin,20+21*font->pf_defaultsize.x,
                                 65,PIX_SRC,NULL,'_');
                         nbcar = 0;
                      }
                      break;

            case 19 : pw_char(fen_pixwin,
                              20+(45+nbcar)*font->pf_defaultsize.x,
                              45,PIX_SRC,NULL,' ');
                      *(sup + nbcar) = '\0';

/** si le parametre est egal a 's' -> impression de 'service' a la  **/
/** place de 's' et impression des libelles des parametres suivants **/
                      if (strcmp(sup,"s\0") == 0) {
                         pw_text(fen_pixwin,20+45*font->pf_defaultsize.x,
                                 45,PIX_SRC,NULL,"service");
                         pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                          "identifiant du service                    : _");
                         pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                          "identifiant du nouveau patron du service  :  ");
                      nbmot = 20;
```

```
                    }
                    else

/** si le parametre est egal a 'd' -> impression de 'dactylo' a la  **/
/** place de 'd' et impression des libelles des parametres suivants **/
                    if (strcmp(sup,"d\0") == 0) {
                       pw_text(fen_pixwin,20+45*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"dactylo");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                        "identifiant du nouveau service              : _");
                    nbmot = 22;
                    }
                    else

/** si le parametre est egal a 'se' -> impression de 'secretaire' a **/
/** la place de 'se' et impression des libelles des parametres      **/
/** suivants                                                        **/
                    if (strcmp(sup,"se\0") == 0) {
                       pw_text(fen_pixwin,20+45*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,"secretaire");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                        "identifiant du nouveau service              : _");
                    nbmot = 22;
                    }
                    else {

/** si le parametre n'est egal ni a 's',ni a 'd', **/
/** ni a 'se' -> on efface ce que l'utilisateur a **/
/** tape et on attend qu'il retape ce parametre   **/
                       pw_writebackground(fen_pixwin,
                                          20+45*font->pf_defaultsize.x,
                                          29,500,20,PIX_CLR);
                       pw_char(fen_pixwin,20+45*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,'_');
                    }
                    nbcar = 0;
                    break;

            case 20 : pw_char(fen_pixwin,20+45*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,'_');
                    pw_char(fen_pixwin,
                              20+(45+nbcar)*font->pf_defaultsize.x,
                              65,PIX_SRC,NULL,' ');
                    nbcar = 0;
                    nbmot = 21;
                    break;

            case 21 : pw_char(fen_pixwin,
                              20+(45+nbcar)*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,' ');
                    impmes("  MODIFICATION EFFECTUEE",1);
                    adm = 0;
                    okfen = 1;
                    nbcar = 0;
                    break;

            case 22 : pw_char(fen_pixwin,
                              20+(45+nbcar)*font->pf_defaultsize.x,
                              65,PIX_SRC,NULL,' ');
                    impmes("  MODIFICATION EFFECTUEE",1);
                    adm = 0;
```

```
                        okfen = 1;
                        nbcar = 0;
                        break;

            case 23 : pw_char(fen_pixwin,
                              20+(14+nbcar)*font->pf_defaultsize.x,
                              45,PIX_SRC,NULL,' ');

/** impression dans fen2 de la description d'un abonne **/
                        pw_writebackground(fen2_pixwin,0,0,565,600,PIX_CLR);
                        fpdescr = fopen("descr","r");
                        imprime(fpdescr,1,2);
                        fclose(fpdescr);
                        adm = 0;
                        okfen = 1;
                        nbcar = 0;
                        break;
          }
       }
       else {

/** selon le parametre que l'on est en train de taper, calcul des **/
/** coordonnees de l'emplacement dans fen du prochain caractere a **/
/** imprimer                                                      **/
          switch (nbmot) {

                case 1 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 20;
                         break;

                case 2 : coordx = 20 + (16+nbcar)*font->pf_defaultsize.x;
                         coordy = 40;
                         break;

                case 3 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 20;
                         break;

                case 4 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 40;
                         break;

                case 5 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 60;
                         break;

                case 6 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 80;
                         break;

                case 7 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 40;
                         break;

                case 8 : coordx = 20 + (32+nbcar)*font->pf_defaultsize.x;
                         coordy = 60;
                         break;

                case 9 : coordx = 20 + (23+nbcar)*font->pf_defaultsize.x;
                         coordy = 20;
                         break;
```

```
                case 10 : coordx = 20 + (23+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 11 : coordx = 20 + (25+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                case 12 : coordx = 20 + (25+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 13 : coordx = 20 + (25+nbcar)*font->pf_defaultsize.x;
                          coordy = 60;
                          break;

                case 14 : coordx = 20 + (53+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                case 15 : coordx = 20 + (53+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 16 : coordx = 20 + (53+nbcar)*font->pf_defaultsize.x;
                          coordy = 60;
                          break;

                case 17 : coordx = 20 + (21+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                case 18 : coordx = 20 + (21+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 19 : coordx = 20 + (45+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                case 20 : coordx = 20 + (45+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 21 : coordx = 20 + (45+nbcar)*font->pf_defaultsize.x;
                          coordy = 60;
                          break;

                case 22 : coordx = 20 + (45+nbcar)*font->pf_defaultsize.x;
                          coordy = 40;
                          break;

                case 23 : coordx = 20 + (14+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

        }

/** impression dans fen du caractere que l'utilisateur vient de taper **/
        imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
```

```
                                &efface,nbcar);
            if (efface == 1) nbcar--;
            else
              if (nbcar<30) {
                 *(sup + nbcar) = ievent_fen.ie_code;
                 nbcar++;
              }
         }
}

/*****************************************************************************/
```

```
/*************************************************************************/

/**** lecture des parametres pour un archivage ****/

static trait_archive()
{
int efface,coordx,coordy;


/** si le caractere que l'utilisateur vient de taper dans fen  **/
/** est RETURN -> gestion du parametre que l'utilisateur vient **/
/** de finir de taper                                          **/
      if (ievent_fen.ie_code == 13) {

         switch (nbmot) {

            case 1 : *(sup + nbcar) = '\0';

/** si le parametre est egal a 'r' -> on affiche 'recu' a la place **/
/** de 'r' et on affiche le libelle du parametre suivant           **/
                     if (strcmp(sup,"r\0") == 0) {
                        pw_text(fen_pixwin,20+30*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,"recu");
                        pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                                "identifiant (o/n)          : _");
                        nbmot = 2;
                     }
                     else

/** si le parametre est egal a 'e' -> on affiche 'envoye' a la place **/
/** de 'e' et on affiche le libelle du parametre suivant             **/
                     if (strcmp(sup,"e\0") == 0) {
                        pw_text(fen_pixwin,20+30*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,"envoye");
                        pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                                "identifiant (o/n)          : _");
                        nbmot = 9;
                     }
                     else {

/** si le parametre n'est egal ni a 'e', ni a 'r' -> on efface ce que   **/
/** l'utilisateur vient de taper et on attend qu'il retape ce parametre **/
                        pw_writebackground(fen_pixwin,
                                           20+30*font->pf_defaultsize.x,
                                           29,500,20,PIX_CLR);
                        pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,'_');
                     }
                     nbcar = 0;
                     break;

            case 2 : *(sup + nbcar) = '\0';

/** si le parametre est egal a 'o' -> on affiche les libelles des **/
/** parametres suivants                                           **/
                     if (strcmp(sup,"o\0") == 0) {
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                                "expediteur             : _");
                        pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                                "date d'envoi           : ");
                        pw_text(fen_pixwin,20,125,PIX_SRC,NULL,
```

```
                        "numero d'envoi                 : ");
                    pw_char(fen_pixwin,20+31*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,' ');
                    nbmot = 3;
                }
                else

/** si le parametre est egal a 'n' -> on affiche les libelles des **/
/** parametres suivants                                           **/
                if (strcmp(sup,"n\0") == 0) {
                    pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                        "date de reception          : _");
                    pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                        "expediteur                 : ");
                    pw_text(fen_pixwin,20,125,PIX_SRC,NULL,
                        "mots-cle                   : ");
                    pw_char(fen_pixwin,20+31*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,' ');
                    nbmot = 6;
                }
                else {

/** si le parametre n'est egal ni a 'o', ni a 'n' -> on efface ce que   **/
/** l'utilisateur vient de taper et on attend qu'il retape ce parametre **/
                    pw_writebackground(fen_pixwin,
                                       20+30*font->pf_defaultsize.x,
                                       49,500,20,PIX_CLR);
                    pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,'_');
                }
                nbcar = 0;
                break;

            case 3 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,' ');
                pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,'_');
                nbcar = 0;
                nbmot = 4;
                break;

            case 4 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,' ');
                pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        125,PIX_SRC,NULL,'_');
                nbcar = 0;
                nbmot = 5;
                break;

            case 5 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        125,PIX_SRC,NULL,' ');
                nbcar = 0;
                okfen = 1;
                archive = 0;
                suite_archive();
                break;

            case 6 : pw_char(fen_pixwin,
```

```
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,' ');
                pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,'_');
                nbcar = 0;
                nbmot = 7;
                break;

        case 7 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,' ');
                pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        125,PIX_SRC,NULL,'_');
                nbcar = 0;
                nb_mot_cles = 0;
                nbmot = 8;
                break;

        case 8 : if (nbcar == 0) {

/** nbcar = 0 => l'utilisateur a tape directement RETURN pour ce **/
/** parametre -> il a termine de rentrer ses mots-cles           **/
                   okfen = 1;
                   archive = 0;
                   pw_char(fen_pixwin,
                           20+(30+nbcar)*font->pf_defaultsize.x,
                           125,PIX_SRC,NULL,' ');
                   suite_archive();
                }
                else

/** nbcar > 0 => l'utilisateur vient de taper un mot-cle. Si il **/
/** avait deja tape moins de 5 mots-cle, on efface le mot-cle   **/
/** qu'il vient de taper et on attend le suivant                **/
                if (nb_mot_cles < 4) {
                   pw_writebackground(fen_pixwin,
                                      20+30*font->pf_defaultsize.x,
                                      109,500,20,PIX_CLR);
                   pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                           125,PIX_SRC,NULL,'_');
                   nb_mot_cles +=1;
                   nbcar = 0;
                }
                else {

/** l'utilisateur vient de taper son 5eme mot-cle -> liberation de fen **/
                   okfen = 1;
                   archive = 0;
                   pw_char(fen_pixwin,
                           20+(30+nbcar)*font->pf_defaultsize.x,
                           125,PIX_SRC,NULL,' ');
                   suite_archive();
                }
                break;

        case 9 : *(sup + nbcar) = '\0';

/** si le parametre est egal a 'o' -> on affiche les libelles des **/
/** parametres suivants                                           **/
                if (strcmp(sup,"o\0") == 0) {
                   pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
```

```
                        "date d'envoi                    : _");
                    pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                        "numero d'envoi                  : ");
                    pw_char(fen_pixwin,20+31*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,' ');
                    nbmot = 10;
                }
                else

/** si le parametre est egal a 'n' -> on affiche les libelles des **/
/** parametres suivants                                           **/
                if (strcmp(sup,"n\0") == 0) {
                    pw_text(fen_pixwin,20,85,PIX_SRC,NULL,
                        "date d'envoi                    : _");
                    pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                        "destinataire                    : ");
                    pw_text(fen_pixwin,20,125,PIX_SRC,NULL,
                        "mots-cle                        : ");
                    pw_char(fen_pixwin,20+31*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,' ');
                    nbmot = 12;
                }
                else {

/** si le parametre n'est egal ni a 'o', ni a 'n' -> on efface ce que   **/
/** l'utilisateur vient de taper et on attend qu'il retape ce parametre **/
                    pw_writebackground(fen_pixwin,
                                       20+30*font->pf_defaultsize.x,
                                       49,500,20,PIX_CLR);
                    pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,'_');
                }
                nbcar = 0;
                break;

            case 10 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,' ');
                      pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,'_');
                      nbcar = 0;
                      nbmot = 11;
                      break;

            case 11 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,' ');
                      nbcar = 0;
                      okfen = 1;
                      archive = 0;
                      suite_archive();
                      break;

            case 12 : pw_char(fen_pixwin,
                        20+(30+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,' ');
                      pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,'_');
                      nbcar = 0;
                      nbmot = 13;
                      break;
```

```
            case 13 : pw_char(fen_pixwin,
                              20+(30+nbcar)*font->pf_defaultsize.x,
                              105,PIX_SRC,NULL,' ');
                      pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                              125,PIX_SRC,NULL,'_');
                      nbcar = 0;
                      nb_mot_cles = 0;
                      nbmot = 14;
                      break;

            case 14 : if (nbcar == 0) {

/** nbcar = 0 => l'utilisateur a tape directement RETURN pour ce **/
/** parametre -> il a termine de rentrer ses mots-cles           **/
                         okfen = 1;
                         archive = 0;
                         pw_char(fen_pixwin,
                                 20+(30+nbcar)*font->pf_defaultsize.x,
                                 125,PIX_SRC,NULL,' ');
                         suite_archive();
                      }
                      else

/** nbcar > 0 => l'utilisateur vient de taper un mot-cle. Si il **/
/** avait deja tape moins de 5 mots-cle, on efface le mot-cle   **/
/** qu'il vient de taper et on attend le suivant                **/
                      if (nb_mot_cles < 4) {
                         pw_writebackground(fen_pixwin,
                                            20+30*font->pf_defaultsize.x,
                                            109,500,20,PIX_CLR);
                         pw_char(fen_pixwin,20+30*font->pf_defaultsize.x,
                                 125,PIX_SRC,NULL,'_');
                         nb_mot_cles +=1;
                         nbcar = 0;
                      }
                      else {

/** l'utilisateur vient de taper son 5eme mot-cle -> liberation de fen **/
                         okfen = 1;
                         archive = 0;
                         pw_char(fen_pixwin,
                                 20+(30+nbcar)*font->pf_defaultsize.x,
                                 125,PIX_SRC,NULL,' ');
                         suite_archive();
                      }
                      break;
         }
      }
      else {

/** le caractere que l'utilisateur vient de taper n'est pas RETURN -> **/
/** on calcule les coordonnees auxquelles on va imprimer ce caractere **/
/** selon le numero de parametre dont il fait partie                  **/
         coordx = 20 + (30+nbcar)*font->pf_defaultsize.x;
         switch (nbmot) {

                case 1 : coordy = 20;
                         break;

                case 2 : coordy = 40;
```

```
                        break;

                case 3 : coordy = 60;
                        break;

                case 4 : coordy = 80;
                        break;

                case 5 : coordy = 100;
                        break;

                case 6 : coordy = 60;
                        break;

                case 7 : coordy = 80;
                        break;

                case 8 : coordy = 100;
                        break;

                case 9 : coordy = 40;
                        break;

                case 10 : coordy = 60;
                        break;

                case 11 : coordy = 80;
                        break;

                case 12 : coordy = 60;
                        break;

                case 13 : coordy = 80;
                        break;

                case 14 : coordy = 100;
                        break;
        }

/** impression du caractere qui vient d'etre tape dans fen **/
        imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                &efface,nbcar);
        if (efface == 1) nbcar--;
        else
          if (nbcar<30) {
             *(sup + nbcar) = ievent_fen.ie_code;
             nbcar++;
          }
      }
}

/*****************************************************************/

/**** initialise la consultation des documents archives ****/

static suite_archive()
{
      if ((nbmot == 5) || (nbmot == 11)) {

/** si, selon la reponse donnee aux parametres d'un archivage, on   **/
/** consulte un document archive -> affichage dans fen2 du document **/
```

```
/** archive (fichier arch)                                          **/
        fparch = fopen("arch","r");
        imprime(fparch,1,2);
        fclose(fparch);
        arch_uniq = 1;
    }
    else {

/** sinon -> affichage du premier resume de la collection des **/
/** documents archives dans fen3 et du premier document de la **/
/** collection des documents archives dans fen2                **/
        fpres = fopen("res","r");
        imprime(fpres,1,3);
        fclose(fpres);
        fpdoc = fopen("doc","r");
        imprime(fpdoc,1,2);
        fclose(fpdoc);
        arch_uniq = 0;
    }
}

/*****************************************************************/

/**** lecture des parametres d'envoi d'un rappel ****/

static trait_att_rep()
{
int efface,coordx,coordy;


/** si le caractere que l'utilisateur vient de taper dans fen  **/
/** est RETURN -> gestion du parametre que l'utilisateur vient **/
/** de finir de taper                                          **/
    if (ievent_fen.ie_code == 13) {

        switch (nbmot) {

            case 1 : pw_char(fen_pixwin,
                             30+(21+nbcar)*font->pf_defaultsize.x,
                             55,PIX_SRC,NULL,' ');
                     pw_char(fen_pixwin,
                             30+21*font->pf_defaultsize.x,
                             75,PIX_SRC,NULL,'_');
                     nbcar = 0;
                     nbmot = 2;
                     break;

            case 2 : *(sup + nbcar) = '\0';

/** si le parametre est egal a 'o' ou est egal a 'n' -> passage **/
/** au parametre suivant                                        **/
                     if ((strcmp(sup,"o\0")==0) ||
                         (strcmp(sup,"n\0")==0)) {
                        pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                                95,PIX_SRC,NULL,'_');
                        pw_char(fen_pixwin,
                                30+22*font->pf_defaultsize.x,
                                75,PIX_SRC,NULL,' ');
                        nbmot = 3;
                     }
                     else {
```

```
/** si le parametre n'est egal ni a 'o', ni a 'n' -> on efface ce **/
/** que l'utilisateur vient de taper et on attend qu'il retape ce **/
/** parametre                                                     **/
                        pw_writebackground(fen_pixwin,
                                30+21*font->pf_defaultsize.x,59,
                                440,20,PIX_CLR);
                        pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                                75,PIX_SRC,NULL,'_');
                    }
                    nbcar = 0;
                    break;

            case 3 : *(sup + nbcar) = '\0';

/** si le parametre est egal a 'o' ou est egal a 'n' -> passage **/
/** au parametre suivant                                        **/
                    if ((strcmp(sup,"o\0")==0) ||
                        (strcmp(sup,"n\0")==0)) {
                       pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                               115,PIX_SRC,NULL,'_');
                       pw_char(fen_pixwin,
                               30+22*font->pf_defaultsize.x,
                               95,PIX_SRC,NULL,' ');
                       nbmot = 4;
                    }
                    else {

/** si le parametre n'est egal ni a 'o', ni a 'n' -> on efface ce **/
/** que l'utilisateur vient de taper et on attend qu'il retape ce **/
/** parametre                                                     **/
                        pw_writebackground(fen_pixwin,
                                30+21*font->pf_defaultsize.x,79,
                                440,20,PIX_CLR);
                        pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                                95,PIX_SRC,NULL,'_');
                    }
                    nbcar = 0;
                    break;

            case 4 : pw_char(fen_pixwin,
                             30+(21+nbcar)*font->pf_defaultsize.x,
                             115,PIX_SRC,NULL,' ');
                     pw_char(fen_pixwin,
                             30+21*font->pf_defaultsize.x,
                             125,PIX_SRC,NULL,'_');
                     nbcar = 0;
                     nbmot = 5;
                     nb_mot_cles = 0;
                     break;

            case 5 : if (nbcar == 0) {

/** nbcar = 0 => l'utilisateur a tape directement RETURN pour ce **/
/** parametre -> il a termine de rentrer ses mots-cles          **/
                        okfen = 1;
                        att_rep = 0;
                        pw_char(fen_pixwin,
                                30+(21+nbcar)*font->pf_defaultsize.x,
                                125,PIX_SRC,NULL,' ');
                        impmes("  RAPPEL ENVOYE",1);
```

```
                    }
                    else {

/** nbcar > 0 => l'utilisateur vient de taper un mot-cle. Si il **/
/** avait deja tape moins de 5 mots-cle, on efface le mot-cle   **/
/** qu'il vient de taper et on attend le suivant                **/
                        if (nb_mot_cles < 4) {
                           pw_writebackground(fen_pixwin,
                                    30+21*font->pf_defaultsize.x,119,
                                    440,20,PIX_CLR);
                           pw_char(fen_pixwin,
                                   30+21*font->pf_defaultsize.x,
                                   135,PIX_SRC,NULL,'_');
                           nb_mot_cles +=1;
                           nbcar = 0;
                        }
                        else {

/** l'utilisateur vient de taper son 5eme mot-cle -> liberation de fen **/
                           okfen = 1;
                           att_rep = 0;
                           pw_char(fen_pixwin,
                                   30+(21+nbcar)*font->pf_defaultsize.x,
                                   135,PIX_SRC,NULL,' ');
                           impmes("  RAPPEL ENVOYE",1);
                        }
                    }
                    break;

            }
        }
        else {

/** le caractere que l'utilisateur vient de taper n'est pas RETURN -> **/
/** on calcule les coordonnees auxquelles on va imprimer ce caractere **/
/** selon le numero de parametre dont il fait partie                  **/
            coordx = 30 + (21+nbcar)*font->pf_defaultsize.x;
            coordy = 30 + (nbmot-1)*20;

/** impression du caractere qui vient d'etre tape dans fen **/
            imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                 &efface,nbcar);
            if (efface == 1) nbcar--;
            else
              if (nbcar<30) {
                 *(sup + nbcar) = ievent_fen.ie_code;
                 nbcar++;
              }
        }
}

/*****************************************************************/

/**** gestion du changement des mots de passe ****/

static trait_mot_passe()
{
int efface,coordx,coordy;


/** si le caractere que l'utilisateur vient de taper dans fen  **/
```

```
/** est RETURN -> gestion du parametre que l'utilisateur vient **/
/** de finir de taper                                          **/
        if (ievent_fen.ie_code == 13) {

              if ((nbmot==1) || (nbmot==2) || (nbmot==4) || (nbmot==5))
                 *(new_mot_passe1 + nbcar) = '\0';
              else *(new_mot_passe2 + nbcar) = '\0';


/** si l'utilisateur tape directement RETURN pour le premier **/
/** parametre -> on passe directement au mot de passe de     **/
/** confidentialite                                          **/
              if ((nbcar == 0) && (nbmot == 1)) {
                 pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                 pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                         "mot de passe de confidentialite : ");
                 pw_char(fen_pixwin,
                         30 + 35*font->pf_defaultsize.x,
                         45,PIX_SRC,NULL,'_');
                 newnbmot = 4;
              }

/** si l'utilisateur n'a pas tape directement RETURN pour le premier **/
/** parametre -> traitement de ce parametre                          **/
              if ((nbcar != 0) && (nbmot == 1)) {

/** si le parametre est egal au mot de passe de l'utilisateur -> **/
/** preparation de fen pour la demande du nouveau mot de passe   **/
                 if (strcmp(mot_passe,new_mot_passe1) == 0) {
                    pw_text(fen_pixwin,20,75,PIX_SRC,NULL,
                            "nouveau mot de passe : ");
                    pw_char(fen_pixwin,
                            30 + 24*font->pf_defaultsize.x,
                            75,PIX_SRC,NULL,'_');
                    pw_char(fen_pixwin,
                            30 + (24+nbcar)*font->pf_defaultsize.x,
                            45,PIX_SRC,NULL,' ');
                    nb_mot_passe_false = 0;
                    newnbmot = 2;
                 }
                 else {

/** si le parametre est different du mot de passe -> message d'erreur **/
/** puis test pour ne permettre que 5 essais                          **/
                    impmes("  MOT DE PASSE INCORRECT",1);

/** si l'utilisateur n'a pas encore essaye 5 fois -> preparation **/
/** de fen pour la fois suivante                                 **/
                    if (nb_mot_passe_false < 4) {
                       pw_writebackground(fen_pixwin,0,26,900,149,
                                          PIX_CLR);
                       pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                               "mot de passe d'abonne : ");
                       pw_char(fen_pixwin,
                               30 + 24*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,'_');
                       nb_mot_passe_false +=1;
                    }
                    else {

/** si l'utilisateur a deja essaye 5 fois -> liberation de fen **/
```

```
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        okfen = 1;
                        mp = 0;
                    }
                }
            }

            if (nbmot == 2) {
                pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                        "nouveau mot de passe : ");
                pw_char(fen_pixwin,
                        30 + 24*font->pf_defaultsize.x,
                        105,PIX_SRC,NULL,'_');
                pw_char(fen_pixwin,
                        30 + (24+nbcar)*font->pf_defaultsize.x,
                        75,PIX_SRC,NULL,' ');
                newnbmot = 3;
            }

            if (nbmot == 3) {

/** si le nouveau mot de passe et sa confirmation sont egaux -> **/
/** changement du mot de passe et preparation de fen pour le    **/
/** mot de passe de confidentialite                             **/
                if (strcmp(new_mot_passe1,new_mot_passe2) == 0) {
                    strcpy(mot_passe,new_mot_passe1);
                    pw_writebackground(fen_pixwin,0,26,900,149,
                                       PIX_CLR);
                    pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                            "mot de passe de confidentialite : ");
                    pw_char(fen_pixwin,
                            30 + 35*font->pf_defaultsize.x,
                            45,PIX_SRC,NULL,'_');
                    newnbmot = 4;
                }
                else {

/** si le nouveau mot de passe et sa confirmation sont differents -> **/
/** on recommence a redemander le nouveau mot de passe                **/
                    pw_writebackground(fen_pixwin,0,55,900,94,
                                       PIX_CLR);
                    pw_text(fen_pixwin,20,75,PIX_SRC,NULL,
                            "nouveau mot de passe : ");
                    pw_char(fen_pixwin,
                            30 + 24*font->pf_defaultsize.x,
                            75,PIX_SRC,NULL,'_');
                    newnbmot = 2;
                }
            }

/** si l'utilisateur tape directement RETURN pour le quatrieme **/
/** parametre -> on nettoye et on libere fen                   **/
            if ((nbmot == 4) && (nbcar == 0)) {
                pw_writebackground(fen_pixwin,0,0,900,149,PIX_CLR);
                mp = 0;
                okfen = 1;
            }

            if ((nbmot == 4) && (nbcar != 0)) {
```

```
/** si l'utilisateur n'a pas tape directement RETURN pour le quatrieme **/
/** parametre -> traitement de ce parametre                            **/
                    if (strcmp(mot_passe_confi,new_mot_passe1) == 0) {

/** si le parametre est egal au mot de passe de confidentialite  **/
/** de l'utilisateur -> preparation de fen pour la demande du    **/
/** nouveau mot de passe de confidentialite                      **/
                       pw_text(fen_pixwin,20,75,PIX_SRC,NULL,
                               "nouveau mot de passe : ");
                       pw_char(fen_pixwin,
                               30 + 24*font->pf_defaultsize.x,
                               75,PIX_SRC,NULL,'_');
                       pw_char(fen_pixwin,
                               30 + (35+nbcar)*font->pf_defaultsize.x,
                               45,PIX_SRC,NULL,' ');
                       newnbmot = 5;
                    }
                    else {

/** si le parametre est different du mot de passe -> message d'erreur **/
/** puis test pour ne permettre que 5 essais                          **/
                     impmes("  MOT DE PASSE INCORRECT",1);

/** si l'utilisateur n'a pas encore essaye 5 fois -> preparation **/
/** de fen pour la fois suivante                                 **/
                       if (nb_mot_passe_false < 4) {
                          pw_writebackground(fen_pixwin,0,26,900,149,
                                             PIX_CLR);
                          pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                  "mot de passe de confidentialite : ");
                          pw_char(fen_pixwin,
                                  30 + 35*font->pf_defaultsize.x,
                                  45,PIX_SRC,NULL,'_');
                          nb_mot_passe_false +=1;
                       }
                       else {

/** si l'utilisateur a deja essaye 5 fois -> liberation de fen **/
                          pw_writebackground(fen_pixwin,0,0,900,149,
                                             PIX_CLR);
                          okfen = 1;
                          mp = 0;
                       }
                    }
                 }

                 if (nbmot == 5) {
                    pw_text(fen_pixwin,20,105,PIX_SRC,NULL,
                            "nouveau mot de passe : ");
                    pw_char(fen_pixwin,
                            30 + 24*font->pf_defaultsize.x,
                            105,PIX_SRC,NULL,'_');
                    pw_char(fen_pixwin,
                            30 + (24+nbcar)*font->pf_defaultsize.x,
                            75,PIX_SRC,NULL,' ');
                    newnbmot = 6;
                 }

                 if (nbmot == 6) {

/** si le nouveau mot de passe et sa confirmation sont egaux -> **/
```

```
/** changement du mot de passe de confidentialite et liberation **/
/** de fen                                                      **/
                    if (strcmp(new_mot_passe1,new_mot_passe2) == 0) {
                       strcpy(mot_passe_confi,new_mot_passe1);
                       pw_writebackground(fen_pixwin,0,0,900,149,
                           PIX_CLR);
                       okfen = 1;
                       mp = 0;
                    }
                    else {

/** si le nouveau mot de passe de confidentialite et sa confirmation **/
/** sont differents -> on recommence a redemander le nouveau mot de  **/
/** passe de confidentialite                                         **/
                       pw_writebackground(fen_pixwin,0,55,900,94,
                                          PIX_CLR);
                       pw_text(fen_pixwin,20,75,PIX_SRC,NULL,
                               "nouveau mot de passe : ");
                       pw_char(fen_pixwin,
                               30 + 24*font->pf_defaultsize.x,
                               75,PIX_SRC,NULL,'_');
                       newnbmot = 5;
                    }
                 }
              nbcar = 0;
           }

           else {

/** le caractere que l'utilisateur vient de taper n'est pas RETURN -> **/
/** on calcule les coordonnees auxquelles on va imprimer ce caractere **/
/** selon le numero de parametre dont il fait partie                  **/
              switch (nbmot) {

                 case 1 : coordx = 30 + (24+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                 case 2 : coordx = 30 + (24+nbcar)*font->pf_defaultsize.x;
                          coordy = 50;
                          break;

                 case 3 : coordx = 30 + (24+nbcar)*font->pf_defaultsize.x;
                          coordy = 80;
                          break;

                 case 4 : coordx = 30 + (35+nbcar)*font->pf_defaultsize.x;
                          coordy = 20;
                          break;

                 case 5 : coordx = 30 + (24+nbcar)*font->pf_defaultsize.x;
                          coordy = 50;
                          break;

                 case 6 : coordx = 30 + (24+nbcar)*font->pf_defaultsize.x;
                          coordy = 80;
                          break;
              }


/** on imprime le caractere que l'utilisateur vient de taper dans fen **/
              imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
```

```
                                    &efface,nbcar);
            if (efface == 1)
               nbcar--;
            else
                if (nbcar<30) {
                   if ((nbmot == 1) || (nbmot == 2) || (nbmot == 4)
                       || (nbmot == 5))
                      *(new_mot_passe1 + nbcar) = ievent_fen.ie_code;
                   else *(new_mot_passe2 + nbcar) = ievent_fen.ie_code;
                   ++nbcar;
                }
         }
         nbmot = newnbmot;
}

/*************************************************************************/

/**** gestion des caracteres tapes lors de la creation ****/
/**** d'une liste                                      ****/

static creer_liste()
{
int  i,coordx,coordy,efface,z;
char *text = "nom d'abonne : \0";
char *text2 = "nom de la liste : \0";

         if (ievent_fen.ie_code == 13)

/** le caractere est un return **/
            if (nbcar == 0)

/** le caractere precedent etait un return egalement **/
               fin_fen_input();
            else {

/** le caractere precedent n'etait pas un return -> on est **/
/** a la fin d'un mot tape par l'utilisateur               **/

/** remise a blanc de la messagesubwindow **/
               impmes("   ",1);

/** effacement du curseur se trouvant a la fin du mot precedent **/
               pw_char(fen_pixwin,50 - abonne +
                       (16+nbcar)*font->pf_defaultsize.x,
                       55 + abonne,PIX_SRC,NULL,' ');

/** cloture du tableau ou se trouve le mot **/
               *(lec_fen[num_lec_fen] + nbcar) = '\0';

/** test pour voir si la liste existe **/
               if ((exist_liste(*lec_fen,&z) == 1)
                   && (abonne == 0)) {

/** si le nom demande est un nom de liste existant deja -> message **/
/** d'erreur et effacement du nom tape                             **/
                  impmes("  LISTE EXISTANT DEJA",1);
                  nbcar = 0;
                  pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                  pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text2);
                  pw_char(fen_pixwin,50 + 16*font->pf_defaultsize.x,
                          55,PIX_SRC,NULL,'_');
```

```
                }
                else
                   if ((exist_abonne(lec_fen[num_lec_fen],
                        "liste des abonnes\0")==0) && (abonne==30)) {

/** si le nom demande est un nom d'abonne n'existant pas -> message **/
/** d'erreur et effacement du nom tape                              **/
                      impmes("  ABONNE N'EXISTANT PAS",1);
                      nbcar = 0;
                      pw_writebackground(fen_pixwin,0,65,
                                         900,84,PIX_CLR);
                      pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                      pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,'_');
                   }
                   else {

/** si le nom tape (soit de liste, soit d'abonne) est valide -> **/
/** incrementation du compteur du tableau contenant la nouvelle **/
/** liste et preparation de l'ecran pour la reception du nom    **/
/** suivant                                                     **/
                      nbcar = 0;
                      ++num_lec_fen;
                      abonne = 30;
                      pw_writebackground(fen_pixwin,0,65,
                                         900,84,PIX_CLR);
                      pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                      pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,'_');
                   }
             }
             else {

/** si le caractere tape n'est pas un return -> impression de ce **/
/** caractere et stockage dans le tableau approprie avec         **/
/** incrementation du compteur de caractere                      **/
                coordx = 50 - abonne + (16+nbcar)*font->pf_defaultsize.x;
                coordy = 30 + abonne;
                imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                     &efface,nbcar);
                if (efface == 1)
                   nbcar--;
                else
                  if (nbcar<30) {
                     *(lec_fen[num_lec_fen] + nbcar) = ievent_fen.ie_code;
                     nbcar++;
                  }
             }
}

/*****************************************************************/

/**** gestion des caracteres tapes lors de la suppression ****/
/**** d'une liste                                         ****/

static supprimer_liste()
{
int coordx,coordy,efface;

          if (ievent_fen.ie_code == 13) {
```

```
/** si le caractere est un return -> cloture du tableau et gestion **/
/** de la fin de commande                                          **/
             *(sup + nbcar) = '\0';
             fin_fen_input();
          }
          else {

/** si le caractere n'est pas un return -> impression de ce       **/
/** caractere et stockage dans le tableau approprie avec          **/
/** incrementation du compteur de caractere                       **/
            coordx = 50 + (16+nbcar)*font->pf_defaultsize.x;
            coordy = 30;
            imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                 &efface,nbcar);
            if (efface == 1) nbcar--;
            else
              if (nbcar<30) {
                 *(sup + nbcar) = ievent_fen.ie_code;
                 nbcar++;
              }
            }
}

/*****************************************************************/

/**** gestion des caracteres tapes lors de l'ajout d'abonnes ****/
/**** a une liste                                            ****/

static ajout_abonne()
{
int  i,coordx,coordy,efface,z;
char *text = "nom d'abonne : \0";
char *text2 = "nom de la liste : \0";

        if (ievent_fen.ie_code == 13)

/** le caractere est un return **/
           if (nbcar == 0)

/** le caractere precedent etait un return egalement **/
              fin_fen_input();
           else {

/** le caractere precedent n'etait pas un return -> on est **/
/** a la fin d'un mot tape par l'utilisateur               **/

/** remise a blanc de la messagesubwindow **/
              impmes("   ",1);

/** effacement du curseur se trouvant a la fin du mot precedent **/
              pw_char(fen_pixwin,50 - abonne +
                      (16+nbcar)*font->pf_defaultsize.x,
                      55 + abonne,PIX_SRC,NULL,' ');

/** cloture du tableau ou se trouve le mot **/
              *(lec_fen2[num_lec_fen2] + nbcar) = '\0';

/** test pour voir si la liste existe **/
              if ((exist_liste(*lec_fen2,&z) == 0) && (abonne == 0)) {

/** si le nom demande est un nom de liste n'existant pas -> message **/
```

```
/** d'erreur et effacement du nom tape                                  **/
                impmes("  LISTE N'EXISTANT PAS",1);
                nbcar = 0;
                pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text2);
                pw_char(fen_pixwin,50 + 16*font->pf_defaultsize.x,
                        55,PIX_SRC,NULL,'_');
             }
             else
                if ((strcmp(*lec_fen2,"liste des abonnes\0") == 0)
                    && (abonne == 0)) {

/** si le nom demande est la liste des abonnes -> message d'erreur **/
/** et effacement du nom tape                                      **/
                   impmes("  PAS D'AJOUT D'ABONNES",1);
                   nbcar = 0;
                   pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                   pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text2);
                   pw_char(fen_pixwin,50 + 16*font->pf_defaultsize.x,
                           55,PIX_SRC,NULL,'_');
                }
                else
                   if ((exist_abonne(lec_fen2[num_lec_fen2],
                       "liste des abonnes\0") ==0) && (abonne==30)) {

/** si le nom demande est un nom d'abonne n'existant pas -> message **/
/** d'erreur et effacement du nom tape                             **/
                      impmes("  ABONNE N'EXISTANT PAS",1);
                      nbcar = 0;
                      pw_writebackground(fen_pixwin,0,65,
                                         900,84,PIX_CLR);
                      pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                      pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,'_');
                   }
                   else {

/** si le nom tape (soit de liste, soit d'abonne) est valide -> **/
/** incrementation du compteur du tableau contenant la nouvelle **/
/** liste et preparation de l'ecran pour la reception du nom    **/
/** suivant                                                     **/
                      nbcar = 0;
                      ++num_lec_fen2;
                      abonne = 30;
                      pw_writebackground(fen_pixwin,0,65,900,
                                         84,PIX_CLR);
                      pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                      pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                              85,PIX_SRC,NULL,'_');
                   }
          }
       else {

/** si le caractere tape n'est pas un return -> impression de ce **/
/** caractere et stockage dans le tableau approprie avec         **/
/** incrementation du compteur de caractere                      **/
          coordx = 50 - abonne + (16+nbcar)*font->pf_defaultsize.x;
          coordy = 30 + abonne;
          imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                               &efface,nbcar);
          if (efface == 1) nbcar--;
```

```
            else
               if (nbcar<30) {
                  *(lec_fen2[num_lec_fen2] + nbcar) =ievent_fen.ie_code;
                  nbcar++;
               }
         }
}

/*****************************************************************/

/**** gestion des caracteres tapes lors de la suppression ****/
/**** d'abonnes d'une liste                                ****/

static suppress_abonne()
{
int  i,coordx,coordy,efface,z;
char *text = "nom d'abonne : \0";
char *text2 = "nom de la liste : \0";

        if (ievent_fen.ie_code == 13)

/** le caractere est un return **/
           if (nbcar == 0)

/** le caractere precedent etait un return egalement **/
               fin_fen_input();
           else {

/** le caractere precedent n'etait pas un return -> on est **/
/** a la fin d'un mot tape par l'utilisateur                **/

/** remise a blanc de la messagesubwindow **/
               impmes("    ",1);

/** effacement du curseur se trouvant a la fin du mot precedent **/
               pw_char(fen_pixwin,50 - abonne +
                       (16+nbcar)*font->pf_defaultsize.x,
                       55 + abonne,PIX_SRC,NULL,' ');

/** cloture du tableau ou se trouve le mot **/
               *(lec_fen2[num_lec_fen2] + nbcar) = '\0';

/** test pour voir si la liste existe **/
               if ((exist_liste(*lec_fen2,&z) == 0) && (abonne == 0)) {

/** si le nom demande est un nom de liste n'existant pas -> message **/
/** d'erreur et effacement du nom tape                              **/
                  impmes("  LISTE N'EXISTANT PAS",1);
                  nbcar = 0;
                  pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                  pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text2);
                  pw_char(fen_pixwin,50 + 16*font->pf_defaultsize.x,
                          55,PIX_SRC,NULL,'_');
               }
               else
                  if ((strcmp(*lec_fen2,"liste des abonnes\0") == 0)
                      && (abonne == 0)) {

/** si le nom demande est la liste des abonnes -> message d'erreur **/
/** et effacement du nom tape                                      **/
                     impmes("  PAS DE SUPPRESSION D'ABONNES",1);
```

```
                    nbcar = 0;
                    pw_writebackground(fen_pixwin,0,26,900,149,PIX_CLR);
                    pw_text(fen_pixwin,20,55,PIX_SRC,NULL,text2);
                    pw_char(fen_pixwin,50 + 16*font->pf_defaultsize.x,
                            55,PIX_SRC,NULL,'_');
                }
                else
                    if ((exist_abonne(lec_fen2[num_lec_fen2],
                        *lec_fen2) ==0) && (abonne==30)) {

/** si le nom demande est un nom d'abonne n'existant pas -> message **/
/** d'erreur et effacement du nom tape                              **/
                        impmes("  ABONNE N'EXISTANT PAS",1);
                        nbcar = 0;
                        pw_writebackground(fen_pixwin,0,65,
                                           900,84,PIX_CLR);
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                        pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                                85,PIX_SRC,NULL,'_');
                    }
                    else {

/** si le nom tape (soit de liste, soit d'abonne) est valide -> **/
/** incrementation du compteur du tableau contenant la nouvelle **/
/** liste et preparation de l'ecran pour la reception du nom    **/
/** suivant                                                     **/
                        nbcar = 0;
                        ++num_lec_fen2;
                        abonne = 30;
                        pw_writebackground(fen_pixwin,0,65,900,
                                           84,PIX_CLR);
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text);
                        pw_char(fen_pixwin,20+16*font->pf_defaultsize.x,
                                85,PIX_SRC,NULL,'_');
                    }
            }
        else {

/** si le caractere tape n'est pas un return -> impression de ce **/
/** caractere et stockage dans le tableau approprie avec         **/
/** incrementation du compteur de caractere                      **/
            coordx = 50 - abonne + (16+nbcar)*font->pf_defaultsize.x;
            coordy = 30 + abonne;
            imprim_char_suiv_fen(coordx,coordy,ievent_fen.ie_code,
                                 &efface,nbcar);
            if (efface == 1) nbcar--;
            else
                if (nbcar<30) {
                    *(lec_fen2[num_lec_fen2] + nbcar) =ievent_fen.ie_code;
                    nbcar++;
                }
        }
}

/*****************************************************************/

/**** gestion du nom du ou des destinataires dans la saisie ****/
/**** des parametres d'envoi                                ****/

static saisie_1()
{
```

```
int z;

            if (nbcar == 0) {

/** le nom du destinataire est vide **/
                if (au_moins_un == 1) {

/** l'utilisateur a deja tape au moins un nom de destinataire ->  **/
/** preparation de fen pour la saisie du parametre suivant et     **/
/** incrementation du compteur de parametres                      **/
                    pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,65,
                            PIX_SRC,NULL,'_');
                    nbmot +=1;
                    pw_writebackground(fen_pixwin,
                                       30+21*font->pf_defaultsize.x,29,
                                       440-22*font->pf_defaultsize.x,20,
                                       PIX_CLR);
                }
                else {

/** l'utilisateur n'a pas encore tape de nom valide -> **/
/** message d'erreur                                   **/
                    impmes("  AU MOINS UN DESTINATAIRE",1);
                }
            }
            else {

/** le nom du destinataire n'est pas vide **/
                *(sup + nbcar) = '\0';
                if ((exist_liste(sup,&z)==0) && (exist_abonne(sup,
                    "liste des abonnes\0")==0)) {

/** le destinataire n'est ni une liste ni un abonne existant -> **/
/** message d'erreur                                           **/
                    impmes("  NOM INCONNU",1);
                }
                else {

/** le destinataire existe -> message de confirmation et temoin **/
/** d'existence d'un destinataire valide mis a 1                **/
                    impmes("  NOM ENREGISTRE",1);
                    au_moins_un = 1;
                }

/** preparation de fen pour la saisie du destinataire suivant **/
                nbcar=0;
                pw_writebackground(fen_pixwin,
                                   30+21*font->pf_defaultsize.x,29,
                                   440-22*font->pf_defaultsize.x,20,
                                   PIX_CLR);
                pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,45,
                        PIX_SRC,NULL,'_');
            }
}

/*************************************************************************/

/**** gestion du deuxieme et troisieme parametre lors de ****/
/**** la saisie des parametres d'envoi                   ****/

static saisie_2_3()
```

```
{

/** preparation de fen pour la saisie du parametre suivant et **/
/** incrementation du compteur de parametre                   **/
            pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                    (nbmot+1)*20+25,PIX_SRC,NULL,'_');
            pw_char(fen_pixwin,30+(21+nbcar)*font->pf_defaultsize.x,
                    nbmot*20+25,PIX_SRC,NULL,' ');
            nbmot +=1;
            nbcar = 0;
}

/*************************************************************/

/**** gestion du quatrieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                   ****/

static saisie_4()
{

/** cloture du tableau contenant le quatrieme parametre **/
            *(sup + nbcar) = '\0';

            if ((strcmp(sup,"o\0")==0) || (strcmp(sup,"n\0")==0)) {

/** le quatrieme parametre est valide (= 'o' || 'n') -> **/
/** preparation de fen pour le parametre suivant et     **/
/** incrementation du compteur de parametre             **/
               pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                       (nbmot+1)*20+25,PIX_SRC,NULL,'_');
               pw_char(fen_pixwin, 30+(21+nbcar)*font->pf_defaultsize.x,
                       nbmot*20+25,PIX_SRC,NULL,' ');
               nbmot +=1;
            }
            else {

/** le quatrieme parametre n'est pas valide -> suppression du mot **/
/** tape                                                          **/
               pw_writebackground(fen_pixwin,
                                  30+21*font->pf_defaultsize.x,89,
                                  440-22*font->pf_defaultsize.x,20,
                                  PIX_CLR);
               pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                       nbmot*20+25,PIX_SRC,NULL,'_');
            }
            nbcar = 0;
}
```

```
/*****************************************************************/

/**** gestion du cinquieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                   ****/

static saisie_5()
{
char *text9 = "dem. de rep. (o/n) : \0";

/** cloture du tableau contenant le cinquieme parametre **/
            *(sup + nbcar) = '\0';

            if ((strcmp(sup,"o\0")==0) || (strcmp(sup,"n\0")==0)) {

/** le cinquieme parametre est valide (= 'o' || 'n') -> **/
/** preparation de fen pour le parametre suivant et     **/
/** incrementation du compteur de parametre             **/
               pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                       (nbmot+1)*20+25,PIX_SRC,NULL,'_');

/** si la reponse au cinquieme parametre est non -> nombre   **/
/** de parametres a demander augmente a 9 et impression du   **/
/** libelle du parametre                                     **/
               if (strcmp(sup,"n\0") == 0) {
                  rap = 9;
                  pw_text(fen_pixwin,470,85,PIX_SRC,NULL,text9);
               }
               pw_char(fen_pixwin, 30+(21+nbcar)*font->pf_defaultsize.x,
                       nbmot*20+25,PIX_SRC,NULL,' ');
               nbmot +=1;
            }
            else {

/** le cinquieme parametre n'est pas valide -> suppression du mot **/
/** tape                                                          **/
               pw_writebackground(fen_pixwin,
                                  30+21*font->pf_defaultsize.x,109,
                                  440-22*font->pf_defaultsize.x,20,
                                  PIX_CLR);
               pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                       nbmot*20+25,PIX_SRC,NULL,'_');
            }
            nbcar = 0;
}

/*****************************************************************/

/**** gestion du sixieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                 ****/

static saisie_6()
{

/** cloture du tableau contenant le sixieme parametre **/
            *(sup + nbcar) = '\0';

            if ((strcmp(sup,"o\0")==0) || (strcmp(sup,"n\0")==0)) {

/** le sixieme parametre est valide (= 'o' || 'n') ->   **/
/** preparation de fen pour le parametre suivant et     **/
/** incrementation du compteur de parametre             **/
```

```
                pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                        45,PIX_SRC,NULL,'_');
                pw_char(fen_pixwin, 30+(21+nbcar)*font->pf_defaultsize.x,
                        nbmot*20+25,PIX_SRC,NULL,' ');
                nbmot +=1;
            }
            else {

/** le sixieme parametre n'est pas valide -> suppression du mot **/
/** tape                                                        **/
                pw_writebackground(fen_pixwin,
                                   30+21*font->pf_defaultsize.x,89,
                                   440-22*font->pf_defaultsize.x,20,
                                   PIX_CLR);
                pw_char(fen_pixwin,30+21*font->pf_defaultsize.x,
                        nbmot*20+25,PIX_SRC,NULL,'_');
            }
            nbcar = 0;
}

/*****************************************************************/

/**** gestion du septieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                  ****/

static saisie_7()
{

/** cloture du tableau contenant le septieme parametre **/
            *(sup + nbcar) = '\0';

            if ((strcmp(sup,"o\0")==0) || (strcmp(sup,"n\0")==0)) {

/** le septieme parametre est valide (= 'o' || 'n') ->  **/
/** preparation de fen pour le parametre suivant et     **/
/** incrementation du compteur de parametre             **/
                pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                        45,PIX_SRC,NULL,' ');
                pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,'_');
                nbmot +=1;
            }
            else {

/** le septieme parametre n'est pas valide -> suppression du mot **/
/** tape                                                        **/
                pw_writebackground(fen_pixwin,
                                   480+21*font->pf_defaultsize.x,29,
                                   440-22*font->pf_defaultsize.x,20,
                                   PIX_CLR);
                pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                        45,PIX_SRC,NULL,'_');
            }
            nbcar = 0;
}

/*****************************************************************/

/**** gestion du huitieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                  ****/
```

```
static saisie_8()
{
            if (nbcar == 0) {
               pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                       65,PIX_SRC,NULL,' ');
               nbmot +=1;

/** si tous les parametres ont ete donnes -> message de fin de **/
/** transfert, reinitialisation du temoin de presence dans la  **/
/** phase de saisie des parametres d'envoi et du temoin de     **/
/** presence d'au moins un destinataire valide                 **/
               if (rap == 8) {
                  impmes("  TRANSFERT EFFECTUE",1);
                  okfen = 1;
                  saisie_param = 0;
                  au_moins_un = 0;
               }
               else
               pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                       85,PIX_SRC,NULL,'_');
            }
            else {
             if (nb_mot_cles < 4) {
                pw_writebackground(fen_pixwin,
                                   480+21*font->pf_defaultsize.x,49,
                                   440,20,PIX_CLR);
                pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,'_');
                nb_mot_cles +=1;
                nbcar = 0;
            }
             else {
                pw_char(fen_pixwin,
                        480+(21+nbcar)*font->pf_defaultsize.x,
                        65,PIX_SRC,NULL,' ');
                nbmot +=1;

/** si tous les parametres ont ete donnes -> message de fin de **/
/** transfert, reinitialisation du temoin de presence dans la  **/
/** phase de saisie des parametres d'envoi et du temoin de     **/
/** presence d'au moins un destinataire valide                 **/
               if (rap == 8) {
                  impmes("  TRANSFERT EFFECTUE",1);
                  okfen = 1;
                  saisie_param = 0;
                  au_moins_un = 0;
               }
               else
                pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,'_');
             }
            }

}

/*****************************************************************/

/**** gestion du neuvieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                  ****/

static saisie_9()
```

```
{
char *text10 = "nom d'expediteur    : \0";
char *text11 = "date d'envoi        : \0";
char *text12 = "numero              : \0";

/** cloture du tableau contenant le neuvieme parametre **/
            *(sup + nbcar) = '\0';

            if ((strcmp(sup,"o\0")==0) || (strcmp(sup,"n\0")==0)) {

/** le neuvieme parametre est valide (= 'o' || 'n') ->  **/
/** preparation de fen pour le parametre suivant et     **/
/** incrementation du compteur de parametre             **/

/** si la reponse au neuvieme parametre est oui -> nombre     **/
/** de parametres a demander augmente a 12 et impression du   **/
/** libelle des parametres                                    **/
                if (strcmp(sup,"o\0") == 0) {
                   pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                           105,PIX_SRC,NULL,'_');
                   pw_text(fen_pixwin,470,105,PIX_SRC,NULL,text10);
                   pw_text(fen_pixwin,470,125,PIX_SRC,NULL,text11);
                   pw_text(fen_pixwin,470,145,PIX_SRC,NULL,text12);
                }
                else {

/** si tous les parametres ont ete donnes -> message de fin de **/
/** transfert, reinitialisation du temoin de presence dans la  **/
/** phase de saisie des parametres d'envoi et du temoin de     **/
/** presence d'au moins un destinataire valide                 **/
                   impmes("  TRANSFERT EFFECTUE",1);
                   okfen = 1;
                   saisie_param = 0;
                   au_moins_un = 0;
                }
                pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                        85,PIX_SRC,NULL,' ');
                nbmot +=1;
            }
            else {

/** le neuvieme parametre n'est pas valide -> suppression du mot **/
/** tape                                                         **/
                pw_writebackground(fen_pixwin,
                                   480+21*font->pf_defaultsize.x,129,
                                   440-22*font->pf_defaultsize.x,20,
                                   PIX_CLR);
                pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                        nbmot*20+25,PIX_SRC,NULL,'_');
            }
            nbcar = 0;
}

/*************************************************************************/

/**** gestion du dixieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                 ****/

static saisie_10()
{
```

```
/** preparation de fen pour la saisie du parametre suivant et **/
/** incrementation du compteur de parametre                   **/
            pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                    125,PIX_SRC,NULL,'_');
            pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                    105,PIX_SRC,NULL,' ');
            nbmot +=1;
            nbcar = 0;
}

/*****************************************************************/

/**** gestion du onzieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                 ****/

static saisie_11()
{

/** preparation de fen pour la saisie du parametre suivant et **/
/** incrementation du compteur de parametre                   **/
            pw_char(fen_pixwin,480+21*font->pf_defaultsize.x,
                    145,PIX_SRC,NULL,'_');
            pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                    125,PIX_SRC,NULL,' ');
            nbmot +=1;
            nbcar = 0;
}

/*****************************************************************/

/**** gestion du douzieme parametre lors de la saisie des ****/
/**** parametres d'envoi                                  ****/

static saisie_12()
{

/** tous les parametres ont ete donnes -> message de fin de    **/
/** transfert, reinitialisation du temoin de presence dans la  **/
/** phase de saisie des parametres d'envoi et du temoin de     **/
/** presence d'au moins un destinataire valide                 **/
            pw_char(fen_pixwin,480+(21+nbcar)*font->pf_defaultsize.x,
                    145,PIX_SRC,NULL,' ');
            nbmot +=1;
            nbcar = 0;
            saisie_param = 0;
            au_moins_un = 0;
            impmes("  TRANSFERT EFFECTUE",1);
            okfen = 1;
}

/*****************************************************************/

/**** cette fonction est appelee asynchronement chaque fois ****/
/**** qu'un bouton de la souris est enfonce dans fen2 .     ****/

static fen2_input(sw,ibits,ebits,obits,timer)
caddr_t sw;
int *ibits,*ebits,*obits;
struct timeval **timer;

{
```

```
int i;

/** lecture de l'evenement (un des trois boutons de la souris) **/
       input_readevent(fen2->ts_windowfd,&ievent_fen2);
       *ibits = *obits = *ebits = 0;
       switch (ievent_fen2.ie_code)  {

        case MS_MIDDLE :

/** bouton du milieu **/
                        if (preslist == 1) {

/** si la liste des listes est dans fen2 -> saisie de l'information **/
/** designee par la souris et impression dans fen2 de la liste      **/
/** correspondant a cette information                               **/
                            get_tab_cont_fen2(ievent_fen2.ie_locx,
                                              ievent_fen2.ie_locy);
                            imprim_liste();
                        }
                        break;

        case MS_RIGHT  :

/** bouton de droite **/
                        if ((pass_icone == 2)
                            && (choice_menu1_fen2 == 'x'))

/** si on est dans bottin (icone 2) et que aucune commande du menu  **/
/** n'est activee pour l'instant -> gestion du menu de consultation **/
/** du bottin                                                       **/
                            ges_menu1_fen2();

/** si on n'est pas dans bottin (icone 2), qu'on est quand meme     **/
/** quelque part, que le document pere n'est pas present a l'ecran, **/
/** qu'on n'est pas dans une consultation de document en attente,   **/
/** et qu'on n'est pas dans consultation administration -> gestion  **/
/** du menu general de consultation                                 **/
                        if ((pass_icone != 2) && (pass_icone != -1)
                            && (pere_pres_res == 0) && (attente != 1)
                            && (pass_icone != 13))
                           ges_menu_cons_doc();
                        break;

        case MS_LEFT   :

/** bouton de gauche **/
                        if ((pass_icone == 2)
                            && (choice_menu1_fen2 != 'x')) {

/** si on est dans bottin (icone 2) et que une commande du menu est **/
/** activee pour l'instant -> saisie de l'information designee par  **/
/** la souris et impression dans fen2 de cette information          **/
                           win_setmouseposition(fen->ts_windowfd,
                                                850,50);
                           get_tab_cont_fen2(ievent_fen2.ie_locx,
                                             ievent_fen2.ie_locy);
                           if (choice_menu1_fen2 == 's')

/** on est dans une suppression de liste **/
                              imprim_clic(sup,50+16*
                                          font->pf_defaultsize.x,30);
```

```
                        if (choice_menu1_fen2 == 'c')

/** on est dans une creation de liste    **/
                            imprim_clic(lec_fen[num_lec_fen],
                                        50-abonne+16*
                                        font->pf_defaultsize.x,
                                        30+abonne);
                        if ((choice_menu1_fen2 == 'a') ||
                            (choice_menu1_fen2 == 'd'))

/** on est dans l'ajout ou la suppression d'un abonne dans une liste **/
                            imprim_clic(lec_fen2[num_lec_fen2],
                                        50-abonne+16*
                                        font->pf_defaultsize.x,
                                        30+abonne);
                    }
                    if ((saisie_param == 1) && (nbmot == 1)) {

/** si on est dans un transfert vers la boite out (icone 1 et **/
/** commande 2) et que l'on est dans la saisie du premier     **/
/** parametre -> saisie de l'information designee par la      **/
/** souris et impression dans fen de cette information        **/
                        win_setmouseposition(fen->ts_windowfd,
                                             850,50);
                        get_tab_cont_fen2(ievent_fen2.ie_locx,
                                          ievent_fen2.ie_locy);
                        imprim_clic(sup,30+21*
                                    font->pf_defaultsize.x,20);
                    }
/** reinitialisation d'info_select **/
                    strcpy(info_select,"\0");
                    break;
        }
}

/*****************************************************************/

/**** imprime dans fen le contenu d'info_select a partir du point ****/
/**** (x,y) et met info_select dans chaine                        ****/

static imprim_clic(chaine,x,y)
char chaine[];
int x,y;
{
int i;

/** remise a blanc de l'endroit ou l'on va imprimer le contenu d'  **/
/** info_select                                                    **/
            for (i=0;i<31;++i)
                pw_char(fen_pixwin,x+i*font->pf_defaultsize.x,
                        y+25,PIX_SRC,NULL,' ');

/** impression du contenu d'info_select dans fen  **/
            for (i=0;i<strlen(info_select);++i) {
                chaine[i] = info_select[i];
                pw_char(fen_pixwin,x+i*font->pf_defaultsize.x,
                        y+25,PIX_SRC,NULL,chaine[i]);
            }
            nbcar = i;
            pw_char(fen_pixwin,x+i*font->pf_defaultsize.x,
                    y+25,PIX_SRC,NULL,'_');
```

```
}

/*****************************************************************/

/**** gere le menu des consultations des resumes ****/

static ges_menu_cons_res()
{
struct rect rr1,rr2;

       switch (pass_icone) {

/** on est dans une consultation archive -> impression du **/
/** menu pour consultation archives et saisie du choix de **/
/** l'utilisateur                                         **/
              case 4 : if (arch_uniq == 0) {
                          if (ievent_fen3.ie_code ==
                             MENU_BUT && win_inputposevent(&ievent_fen3)
                             && (menu_cons_rep_rec =
                                 menu_display(&menu_cons_rep_rec_ptr,
                                              &ievent_fen3,
                                              fen3->ts_windowfd)))
                              choice_menu_cons =
                                        (char) menu_cons_rep_rec->mi_data;
                       }
                       break;

/** on est dans une consultation des documents en attente de **/
/** reponse  -> impression du menu pour consultation des     **/
/** documents en attente de reponse et saisie du choix de    **/
/** l'utilisateur                                            **/
              case 52: if (ievent_fen3.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen3)
                         && (menu_cons_att =
                             menu_display(&menu_cons_att_ptr,
                                          &ievent_fen3,
                                          fen3->ts_windowfd)))
                       choice_menu_cons = (char) menu_cons_att->mi_data;
                       break;

/** on est dans une consultation des documents en attente de  **/
/** traitement -> impression du menu pour consultation des    **/
/** documents en attente de traitement et saisie du choix de  **/
/** l'utilisateur                                             **/
              case 53: if (ievent_fen3.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen3)
                         && (menu_cons_att_tr =
                             menu_display(&menu_cons_att_tr_ptr,
                                          &ievent_fen3,
                                          fen3->ts_windowfd)))
                       choice_menu_cons=(char) menu_cons_att_tr->mi_data;
                 break;

/** on est dans une consultation du signataire -> impression  **/
/** pour une consultation du signataire et saisie du choix de **/
/** l'utilisateur                                             **/
              case 8 : menu_ges_sign_body.m_next = (struct menu *)NULL;
                       menu_cons_sign_body.m_next = &menu_ges_sign_body;
                       menu_cons_sign_ptr = &menu_cons_sign_body;
                       if (ievent_fen3.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen3)
                         && (menu_cons_sign =
```

```
                                menu_display(&menu_cons_sign_ptr,
                                             &ievent_fen3,
                                             fen3->ts_windowfd)))
                            choice_menu_cons = (char) menu_cons_sign->mi_data;
                            break;

/** on est dans une consultation des reponses recues -> impression **/
/** du menu pour consultation des reponses recues et saisie du     **/
/** choix de l'utilisateur                                         **/
                case 11: if (ievent_fen3.ie_code ==
                             MENU_BUT && win_inputposevent(&ievent_fen3)
                             && (menu_cons_rep_rec =
                                 menu_display(&menu_cons_rep_rec_ptr,
                                             &ievent_fen3,
                                             fen3->ts_windowfd)))
                            choice_menu_cons =
                                   (char) menu_cons_rep_rec->mi_data;
                            break;

                default :

/** on est dans une consultation generale -> impression du menu **/
/** general de consultation et saisie du choix de l'utilisateur **/
                            if (ievent_fen3.ie_code ==
                                MENU_BUT && win_inputposevent(&ievent_fen3)
                                && (menu_cons = menu_display(&menu_cons_ptr,
                                                             &ievent_fen3,
                                                        fen3->ts_windowfd)))
                            choice_menu_cons = (char) menu_cons->mi_data;
                            break;
        }

         switch (choice_menu_cons) {

/** l'utilisateur a choisi la commande-menu "premier" -> impression **/
/** dans fen3 du premier resume de la collection qu'il consulte     **/
/** et reinitialisation du choix de l'utilisateur                   **/
                 case 'p' : fpres = fopen("res","r");
                            pw_writebackground(fen3_pixwin,0,0,570,600,
                                               PIX_CLR);
                            imprime(fpres,1,3);
                            fclose(fpres);
                            numres = 1;
                            choice_menu_cons = 'x';
                            if (pere_pres_res == 1) {
                               pere_pres_res = 0;
                               fpdoc = fopen("doc","r");
                               pw_writebackground(fen2_pixwin,0,0,565,600,
                                                  PIX_CLR);
                               imprime(fpdoc,numdoc,2);
                               fclose(fpdoc);
                            }
                            break;

/** l'utilisateur a choisi la commande-menu "dernier" -> impression **/
/** dans fen3 du dernier resume de la collection qu'il consulte     **/
/** et reinitialisation du choix de l'utilisateur                   **/
                 case 'd' : fpres = fopen("res","r");
                            pw_writebackground(fen3_pixwin,0,0,570,600,
                                               PIX_CLR);
                            imprime(fpres,5,3);
```

```
                        fclose(fpres);
                        numres = 5;
                        choice_menu_cons = 'x';
                        if (pere_pres_res == 1) {
                           pere_pres_res = 0;
                           fpdoc = fopen("doc","r");
                           pw_writebackground(fen2_pixwin,0,0,565,600,
                                              PIX_CLR);
                           imprime(fpdoc,numdoc,2);
                           fclose(fpdoc);
                        }
                        break;

/** l'utilisateur a choisi la commande-menu "precedent" -> impression **/
/** dans fen3 du resume precedent celui a l'ecran dans la collection  **/
/** consultee et reinitialisation du choix de l'utilisateur           **/
                case 'a' : fpres = fopen("res","r");
                           pw_writebackground(fen3_pixwin,0,0,570,600,
                                              PIX_CLR);
                           --numres;
                           if (numres == 0) numres = 5;
                           imprime(fpres,numres,3);
                           fclose(fpres);
                           choice_menu_cons = 'x';
                           if (pere_pres_res == 1) {
                              pere_pres_res = 0;
                              fpdoc = fopen("doc","r");
                              pw_writebackground(fen2_pixwin,0,0,565,600,
                                                 PIX_CLR);
                              imprime(fpdoc,numdoc,2);
                              fclose(fpdoc);
                           }
                           break;

/** l'utilisateur a choisi la commande-menu "suivant" -> impression **/
/** dans fen3 du resume suivant celui a l'ecran dans la collection  **/
/** consultee et reinitialisation du choix de l'utilisateur         **/
                case 'v' : fpres = fopen("res","r");
                           pw_writebackground(fen3_pixwin,0,0,570,600,
                                              PIX_CLR);
                           ++numres;
                           if (numres == 6) numres = 1;
                           imprime(fpres,numres,3);
                           fclose(fpres);
                           choice_menu_cons = 'x';
                           if (pere_pres_res == 1) {
                              pere_pres_res = 0;
                              fpdoc = fopen("doc","r");
                              pw_writebackground(fen2_pixwin,0,0,565,600,
                                                 PIX_CLR);
                              imprime(fpdoc,numdoc,2);
                              fclose(fpdoc);
                           }
                           break;

/** l'utilisateur a choisi la commande-menu "selection" -> le   **/
/** document dont le resume est a l'ecran est selectionne       **/
/** et reinitialisation du choix de l'utilisateur               **/
                case 's' : impmes("  DOCUMENT SELECTIONNE",1);
                           doc_select = numres;
                           put_perm(pass_icone);
                           choice_menu_cons = 'x';
                           break;
```

```
/** l'utilisateur a choisi la commande-menu "supprimer" -> le    **/
/** document dont le resume est a l'ecran est supprime et rei-  **/
/** nitialisation du choix de l'utilisateur                      **/
                case 'u' : impmes("  DOCUMENT SUPPRIME",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "modifier" -> s'il   **/
/** n'y a pas d'autres edition en cours, lancement de l'editeur **/
/** VI pour que l'utilisateur puisse modifier le document dont  **/
/** le resume se trouve a l'ecran et reinitialisation du choix  **/
/** de l'utilisateur                                            **/
                case 'm' : if ((edit_en_cours == 0) &&
                               (edit_rappel == 0)   &&
                               (edit_corps == 0)    &&
                               (edit_rep == 0)      &&
                               (edit_mod == 0)      &&
                               (edit_note == 0)) {
                              fpdoc_mod = fopen("doc_mod","w");
                              fpdoc = fopen("doc","r");
                              imprime(fpdoc,numres,1);
                              fclose(fpdoc);
                              fclose(fpdoc_mod);
                              rect_construct(&rr1,0,0,0,0);
                              rect_construct(&rr2,0,0,0,0);
                              wmgr_forktool("e4","doc_mod 2",&rr1,&rr2,0);
                              edit_mod = 1;
                           }
                           else
                         impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "signer" -> le document **/
/** dont le resume se trouve a l'ecran est signe et reinitiali-     **/
/** sation du choix de l'utilisateur                                **/
                case 'i' : impmes("  DOCUMENT SIGNE",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "mod. note modif." ->   **/
/** si aucune edition n'est en cours, lancement de l'editeur VI     **/
/** avec la note de modification et reinitialisation du choix de    **/
/** l'utilisateur                                                   **/
                case 'o' : if ((edit_en_cours == 0) &&
                               (edit_note == 0)     &&
                               (edit_corps == 0)    &&
                               (edit_rep == 0)      &&
                               (edit_rappel == 0)   &&
                               (edit_mod == 0)) {
                              rect_construct(&rr1,0,0,0,0);
                              rect_construct(&rr2,0,0,0,0);
                              wmgr_forktool("e4","note_modif 2",
                                            &rr1,&rr2,0);
                              edit_note = 1;
                           }
                           else
                         impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;
```

```
/** l'utilisateur a choisi la commande-menu "rappel" -> s'il n'y a **/
/** aucune edition en cours, lancement de l'editeur VI avec le     **/
/** document de rappel et reinitialisation du choix de l'utilisa-  **/
/** teur                                                           **/
                case 'r' : if ((okfen == 1)          &&
                               (edit_en_cours == 0) &&
                               (edit_rappel == 0)    &&
                               (edit_corps == 0)     &&
                               (edit_rep == 0)       &&
                               (edit_mod == 0)       &&
                               (edit_note == 0)) {
                              pw_writebackground(fen_pixwin,0,0,900,149,
                                                 PIX_CLR);
                              okfen = 0;
                              edit_rappel = 1;
                              rect_construct(&rr1,0,0,0,0);
                              rect_construct(&rr2,0,0,0,0);
                              wmgr_forktool("e4","rappel 2",&rr1,&rr2,0);
                           }
                           else
                           impmes("  RAPPEL INTERDIT POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "mod. corps" -> s'il   **/
/** n'y a pas d'edition en cours, lancement de l'editeur VI avec   **/
/** le corps du document dont le resume se trouve a l'ecran et     **/
/** reinitialisation du choix de l'utilisateur                     **/
                case 'z' : if ((okfen == 1)          &&
                               (edit_en_cours == 0) &&
                               (edit_rappel == 0)    &&
                               (edit_mod == 0)       &&
                               (edit_corps == 0)     &&
                               (edit_rep == 0)       &&
                               (edit_note == 0)) {
                              if (pass_icone != 12) {
                                 edit_corps = 1;
                                 fpdoc_corps = fopen("corps","w");
                                 fpdoc = fopen("doc","r");
                                 imprime(fpdoc,numres,4);
                                 fclose(fpdoc);
                                 fclose(fpdoc_corps);
                                 rect_construct(&rr1,0,0,0,0);
                                 rect_construct(&rr2,0,0,0,0);
                                 wmgr_forktool("e4","corps 2",
                                               &rr1,&rr2,0);
                              }
                           }
                           else
                            impmes("  RAPPEL INTERDIT POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "mod. rep." -> s'il n'y **/
/** a aucune edition en cours, lancement de l'editeur VI avec la   **/
/** reponse qu'on veut modifier et reinitialisation du choix de    **/
/** l'utilisateur                                                  **/
                case 'w' : if ((okfen == 1)          &&
                               (edit_en_cours == 0) &&
                               (edit_rappel == 0)    &&
                               (edit_mod == 0)       &&
```

```
                        (edit_corps == 0)     &&
                        (edit_rep == 0)       &&
                        (edit_note == 0)) {
                      edit_rep = 1;
                      rect_construct(&rr1,0,0,0,0);
                      rect_construct(&rr2,0,0,0,0);
                      wmgr_forktool("e4","rep 2",&rr1,&rr2,0);
                    }
                    else
                    impmes("  RAPPEL INTERDIT POUR LE MOMENT",1);
                    choice_menu_cons = 'x';
                    break;

        }
}

/*****************************************************************/

/**** gere le menu des consultations des documents ****/

static ges_menu_cons_doc()
{
struct rect rr1,rr2;

       switch (pass_icone) {

/** on est dans une consultation archives -> impression du menu de **/
/** consultation archives et saisie du choix de l'utilisateur      **/
              case 4 : if (arch_uniq != 0) {
                       if (ievent_fen2.ie_code ==
                          MENU_BUT && win_inputposevent(&ievent_fen2)
                          && (menu_cons_pere =
                              menu_display(&menu_cons_pere_ptr,
                                           &ievent_fen2,
                                           fen2->ts_windowfd)))
                           choice_menu_cons =
                                 (char) menu_cons_pere->mi_data;
                     }
                     else {
                       if (ievent_fen2.ie_code ==
                          MENU_BUT && win_inputposevent(&ievent_fen2)
                          && (menu_cons_rep_rec =
                              menu_display(&menu_cons_rep_rec_ptr,
                                           &ievent_fen2,
                                           fen2->ts_windowfd)))
                           choice_menu_cons =
                                    (char) menu_cons_rep_rec->mi_data;
                     }
                     break;

/** on est dans consultation des documents en attente de reponse -> **/
/** impression du menu de consultation des documents en attente de  **/
/** reponse et saisie du choix de l'utilisateur                     **/
              case 52: if (ievent_fen2.ie_code ==
                        MENU_BUT && win_inputposevent(&ievent_fen2)
                        && (menu_cons_att =
                            menu_display(&menu_cons_att_ptr,
                                         &ievent_fen2,
                                         fen2->ts_windowfd)))
                     choice_menu_cons = (char) menu_cons_att->mi_data;
                     break;
```

```
/** on est dans consultation des documents en attente de traitement **/
/** -> impression du menu de consultation des documents en attente  **/
/** de traitement et saisie du choix de l'utilisateur               **/
              case 53: if (ievent_fen2.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen2)
                         && (menu_cons_att_tr =
                             menu_display(&menu_cons_att_tr_ptr,
                                          &ievent_fen2,
                                          fen2->ts_windowfd)))
                       choice_menu_cons=(char) menu_cons_att_tr->mi_data;
                       break;

/** on est dans consultation du signataire -> impression du menu  **/
/** de consultation du signataire et saisie du choix de l'utili-  **/
/** sateur                                                        **/
              case 8 : menu_ges_sign_body.m_next = (struct menu *)NULL;
                       menu_cons_sign_body.m_next = &menu_ges_sign_body;
                       menu_cons_sign_ptr = &menu_cons_sign_body;
                       if (ievent_fen2.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen2)
                         && (menu_cons_sign =
                             menu_display(&menu_cons_sign_ptr,
                                          &ievent_fen2,
                                          fen2->ts_windowfd)))
                       choice_menu_cons = (char) menu_cons_sign->mi_data;
                 break;

/** on est dans consultation des reponses recues -> impression du  **/
/** menu de consultation des reponses recues et saisie du choix    **/
/** de l'utilisateur                                               **/
              case 11: if (ievent_fen2.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen2)
                         && (menu_cons_rep_rec =
                             menu_display(&menu_cons_rep_rec_ptr,
                                          &ievent_fen2,
                                          fen2->ts_windowfd)))
                       choice_menu_cons =
                              (char) menu_cons_rep_rec->mi_data;
                       break;

/** on est dans consultation du document pere -> impression du     **/
/** menu de consultation du document pere et saisie du choix de    **/
/** l'utilisateur                                                  **/
              case 12: if (ievent_fen2.ie_code ==
                         MENU_BUT && win_inputposevent(&ievent_fen2)
                         && (menu_cons_pere=
                             menu_display(&menu_cons_pere_ptr,
                                          &ievent_fen2,
                                          fen2->ts_windowfd)))
                       choice_menu_cons =
                              (char) menu_cons_pere->mi_data;
                       break;

              default :

/** on est dans une consultation generale ->impression du menu  **/
/** general de consultation et saisie du choix de l'utilisateur **/
                       if (ievent_fen2.ie_code ==
                           MENU_BUT && win_inputposevent(&ievent_fen2)
                           && (menu_cons = menu_display(&menu_cons_ptr,
                                                        &ievent_fen2,
```

```
                                                    fen2->ts_windowfd)))
                choice_menu_cons = (char) menu_cons->mi_data;
                        break;
        }


        switch (choice_menu_cons) {

/** l'utilisateur a choisi la commande-menu "premier" -> impression **/
/** dans fen2 du premier document de la collection qu'il consulte     **/
/** et reinitialisation du choix de l'utilisateur                     **/
                case 'p' : fpdoc = fopen("doc","r");
                           pw_writebackground(fen2_pixwin,0,0,565,600,
                                              PIX_CLR);
                           imprime(fpdoc,1,2);
                           fclose(fpdoc);
                           numdoc = 1;
                           choice_menu_cons = 'x';
                           if (pere_pres_doc == 1) {
                              pere_pres_doc = 0;
                              fpres = fopen("res","r");
                              pw_writebackground(fen3_pixwin,0,0,570,600,
                                                 PIX_CLR);
                              imprime(fpres,numres,3);
                              fclose(fpres);
                           }
                           break;

/** l'utilisateur a choisi la commande-menu "dernier" -> impression **/
/** dans fen2 du dernier document de la collection qu'il consulte     **/
/** et reinitialisation du choix de l'utilisateur                     **/
                case 'd' : fpdoc = fopen("doc","r");
                           pw_writebackground(fen2_pixwin,0,0,565,600,
                                              PIX_CLR);
                           imprime(fpdoc,5,2);
                           fclose(fpdoc);
                           numdoc = 5;
                           choice_menu_cons = 'x';
                           if (pere_pres_doc == 1) {
                              pere_pres_doc = 0;
                              fpres = fopen("res","r");
                              pw_writebackground(fen3_pixwin,0,0,570,600,
                                                 PIX_CLR);
                              imprime(fpres,numres,3);
                              fclose(fpres);
                           }
                           break;

/** l'utilisateur a choisi la commande-menu "precedent" -> impression **/
/** dans fen2 du document precedent celui a l'ecran dans la collec-   **/
/** tion qu'il consulte et reinitialisation du choix de l'utilisateur **/
                case 'a' : fpdoc = fopen("doc","r");
                           pw_writebackground(fen2_pixwin,0,0,565,600,
                                              PIX_CLR);
                           --numdoc;
                           if (numdoc == 0) numdoc = 5;
                           imprime(fpdoc,numdoc,2);
                           fclose(fpdoc);
                           choice_menu_cons = 'x';
                           if (pere_pres_doc == 1) {
                              pere_pres_doc = 0;
                              fpres = fopen("res","r");
```

```
                                pw_writebackground(fen3_pixwin,0,0,570,600,
                                                   PIX_CLR);
                                imprime(fpres,numres,3);
                                fclose(fpres);
                            }
                            break;

/** l'utilisateur a choisi la commande-menu "suivant" -> impression    **/
/** dans fen2 du document suivant celui a l'ecran dans la collection   **/
/** qu'il consulte et reinitialisation du choix de l'utilisateur       **/
                case 'v' : fpdoc = fopen("doc","r");
                           pw_writebackground(fen2_pixwin,0,0,565,600,
                                              PIX_CLR);
                           ++numdoc;
                           if (numdoc == 6) numdoc = 1;
                           imprime(fpdoc,numdoc,2);
                           fclose(fpdoc);
                           choice_menu_cons = 'x';
                           if (pere_pres_doc == 1) {
                              pere_pres_doc = 0;
                              fpres = fopen("res","r");
                              pw_writebackground(fen3_pixwin,0,0,570,600,
                                                 PIX_CLR);
                              imprime(fpres,numres,3);
                              fclose(fpres);
                           }
                           break;

/** l'utilisateur a choisi la commande-menu "selection" -> le          **/
/** document a l'ecran est selectionne et reinitialisation du          **/
/** choix de l'utilisateur                                             **/
                case 's' : impmes("  DOCUMENT SELECTIONNE",1);
                           doc_select = numdoc;
                           put_perm(pass_icone);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "supprimer" -> le          **/
/** document a l'ecran est supprime et reinitialisation du             **/
/** choix de l'utilisateur                                             **/
                case 'u' : impmes("  DOCUMENT SUPPRIME",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "modifier" -> s'il         **/
/** n'y a pas d'autres edition en cours, lancement de l'editeur VI      **/
/** pour que l'utilisateur puisse modifier le document qui se trouve   **/
/** a l'ecran et reinitialisation du choix de l'utilisateur            **/
                case 'm' : if ((edit_en_cours == 0) &&
                               (edit_rappel == 0)   &&
                               (edit_corps == 0)    &&
                               (edit_rep == 0)      &&
                               (edit_mod == 0)      &&
                               (edit_note == 0)) {
                           fpdoc_mod = fopen("doc_mod","w");
                           fpdoc = fopen("doc","r");
                           imprime(fpdoc,numdoc,1);
                           fclose(fpdoc);
                           fclose(fpdoc_mod);
                           rect_construct(&rr1,0,0,0,0);
```

```
                        rect_construct(&rr2,0,0,0,0);
                        wmgr_forktool("e4","doc_mod 3",&rr1,&rr2,0);
                        edit_mod = 1;
                    }
                    else
                    impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                    choice_menu_cons = 'x';
                    break;

/** l'utilisateur a choisi la commande-menu "signer" -> le document **/
/** qui se trouve a l'ecran est signe et reinitialisation du choix  **/
/** de l'utilisateur                                                **/
            case 'i' : impmes("  DOCUMENT SIGNE",1);
                    choice_menu_cons = 'x';
                    break;

/** l'utilisateur a choisi la commande-menu "mod. note modif." ->   **/
/** si aucune edition n'est en cours, lancement de l'editeur VI     **/
/** avec la note de modification et reinitialisation du choix de    **/
/** l'utilisateur                                                   **/
            case 'o' : if ((edit_en_cours == 0) &&
                           (edit_rappel == 0)   &&
                           (edit_corps == 0)    &&
                           (edit_rep == 0)      &&
                           (edit_note == 0)     &&
                           (edit_mod == 0)) {
                        rect_construct(&rr1,0,0,0,0);
                        rect_construct(&rr2,0,0,0,0);
                        wmgr_forktool("e4","note_modif 3",
                                      &rr1,&rr2,0);
                        edit_note = 1;
                    }
                    else
                    impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                    choice_menu_cons = 'x';
                    break;

/** l'utilisateur a choisi la commande-menu "rappel" ->             **/
/** si aucune edition n'est en cours, lancement de l'editeur VI     **/
/** avec le rappel et reinitialisation du choix de                  **/
/** l'utilisateur                                                   **/
            case 'r' : if ((okfen == 1)         &&
                           (edit_en_cours == 0) &&
                           (edit_rappel == 0)   &&
                           (edit_mod == 0)      &&
                           (edit_corps == 0)    &&
                           (edit_rep == 0)      &&
                           (edit_note == 0)) {
                        okfen = 0;
                        edit_rappel = 1;
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        rect_construct(&rr1,0,0,0,0);
                        rect_construct(&rr2,0,0,0,0);
                        wmgr_forktool("e4","rappel 3",&rr1,&rr2,0);
                    }
                    else
                    impmes("  RAPPEL INTERDIT POUR LE MOMENT",1);
                    choice_menu_cons = 'x';
                    break;
```

```
/** l'utilisateur a choisi la commande-menu "mod. corps" ->          **/
/** si aucune edition n'est en cours, lancement de l'editeur VI      **/
/** avec le corps du document a l'ecran et reinitialisation du       **/
/** choix de l'utilisateur                                           **/
                case 'z' : if ((okfen == 1)          &&
                               (edit_en_cours == 0) &&
                               (edit_rappel == 0)    &&
                               (edit_mod == 0)       &&
                               (edit_corps == 0)     &&
                               (edit_rep == 0)       &&
                               (edit_note == 0)) {
                              edit_corps = 1;
                              if (((pass_icone != 12) &&
                                   (pass_icone !=4))     ||
                                  ((pass_icone == 4)  &&
                                   (arch_uniq == 0)))  {
                                 fpdoc_corps = fopen("corps","w");
                                 fpdoc = fopen("doc","r");
                                 imprime(fpdoc,numdoc,4);
                                 fclose(fpdoc);
                                 fclose(fpdoc_corps);
                                 rect_construct(&rr1,0,0,0,0);
                                 rect_construct(&rr2,0,0,0,0);
                                 wmgr_forktool("e4","corps 3",
                                               &rr1,&rr2,0);
                              }
                              else
                                if (pass_icone == 12) {
                                 rect_construct(&rr1,0,0,0,0);
                                 rect_construct(&rr2,0,0,0,0);
                                 wmgr_forktool("e4","pere 3",&rr1,&rr2,0);
                                }
                                else {
                                 rect_construct(&rr1,0,0,0,0);
                                 rect_construct(&rr2,0,0,0,0);
                                 wmgr_forktool("e4","arch 3",&rr1,&rr2,0);
                           }
                          }
                          else
                           impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;

/** l'utilisateur a choisi la commande-menu "mod. rep" ->            **/
/** si aucune edition n'est en cours, lancement de l'editeur VI      **/
/** avec la reponse et reinitialisation du choix de l'utilisateur    **/
                case 'w' : if ((okfen == 1)          &&
                               (edit_en_cours == 0) &&
                               (edit_rappel == 0)    &&
                               (edit_mod == 0)       &&
                               (edit_corps == 0)     &&
                               (edit_rep == 0)       &&
                               (edit_note == 0)) {
                              edit_rep = 1;
                              rect_construct(&rr1,0,0,0,0);
                              rect_construct(&rr2,0,0,0,0);
                              wmgr_forktool("e4","rep 3",&rr1,&rr2,0);
                           }
                           else
                           impmes("  EDITION INTERDITE POUR LE MOMENT",1);
                           choice_menu_cons = 'x';
                           break;
```

```
        }
}

/*****************************************************************/

/**** gere le menu de choix de l'attente ****/

static ges_menu_attente()
{

/** impression du menu et saisie du choix de l'utilisateur **/
       if (ievent_fen3.ie_code ==
           MENU_BUT && win_inputposevent(&ievent_fen3)
           && (menu_attente = menu_display(&menu_attente_ptr,
                                           &ievent_fen3,
                                           fen3->ts_windowfd)))
       choix_attente = (char) menu_attente->mi_data;

       switch (choix_attente) {

/** l'utilisateur a choisi la commande-menu "accuse de reception"   **/
/** -> impression du premier document de la collection des doc-     **/
/** uments en attente d'un accuse de reception et impression du     **/
/** resume de ce document                                           **/
              case 'a' : impmes("  ATTENTE ACCUSE DE RECEPTION",1);
                         attente = 0;
                         pass_icone = 51;
                         preslist = 0;
                         pw_writebackground(fen2_pixwin,0,0,565,600,
                                            PIX_CLR);
                         pw_writebackground(fen3_pixwin,0,0,570,600,
                                            PIX_CLR);
                         fpres = fopen("res","r");
                         imprime(fpres,1,3);
                         fclose(fpres);
                         fpdoc = fopen("doc","r");
                         imprime(fpdoc,1,2);
                         fclose(fpdoc);
                         break;

/** l'utilisateur a choisi la commande-menu "reponse" -> impression **/
/** du premier document de la collection des documents en attente   **/
/** de reponse et impression de son resume                          **/
              case 'r' : impmes("  ATTENTE REPONSE",1);
                         attente = 0;
                         pass_icone = 52;
                         preslist = 0;
                         pw_writebackground(fen2_pixwin,0,0,565,600,
                                            PIX_CLR);
                         pw_writebackground(fen3_pixwin,0,0,570,600,
                                            PIX_CLR);
                         fpres = fopen("res","r");
                         imprime(fpres,1,3);
                         fclose(fpres);
                         fpdoc = fopen("doc","r");
                         imprime(fpdoc,1,2);
                         fclose(fpdoc);
                         break;

/** l'utilisateur a choisi la commande-menu "traitement" ->         **/
/** impression du premier document de la collection des documents **/
```

```
/** en attente de traitement et impression de son resume          **/
            case 't' : impmes("  ATTENTE TRAITEMENT",1);
                       attente = 0;
                       pass_icone = 53;
                       preslist = 0;
                       pw_writebackground(fen2_pixwin,0,0,565,600,
                                          PIX_CLR);
                       pw_writebackground(fen3_pixwin,0,0,570,600,
                                          PIX_CLR);
                       fpres = fopen("res","r");
                       imprime(fpres,1,3);
                       fclose(fpres);
                       fpdoc = fopen("doc","r");
                       imprime(fpdoc,1,2);
                       fclose(fpdoc);
                       break;

       }

/** reinitialisation du choix de l'utilisateur  **/
       choix_attente = 'x';
}

/*****************************************************************/

/**** gere le menu de l'administration ****/

static ges_menu_adm()
{

/** impression du menu et saisie du choix de l'utilisateur  **/
       if (ievent_fen3.ie_code ==
           MENU_BUT && win_inputposevent(&ievent_fen3)
           && (menu_adm = menu_display(&menu_adm_ptr,
                                       &ievent_fen3,
                                       fen3->ts_windowfd)))
       choix_adm = (char) menu_adm->mi_data;

       switch (choix_adm) {

/** l'utilisateur a choisi la commande menu "gestion journaliere" **/
/** -> impression de l'ancienne date et demande de la nouvelle,   **/
/** dans fen si fen n'est pas occupee (si occupee, message        **/
/** d'erreur)                                                     **/
            case 'j' : if (okfen == 1) {
                       pw_writebackground(fen_pixwin,0,0,900,149,
                                          PIX_CLR);
                       titre("gestion journaliere");
                       win_setmouseposition(fen->ts_windowfd,850,50);
                       pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                               "ancienne date : 6/6/1944");
                       pw_text(fen_pixwin,20,65,PIX_SRC,NULL,
                               "nouvelle date : _");
                       nbcar = 0;
                       nbmot = 2;
                       adm = 1;
                       okfen = 0;
                      }
                      else  impmes("  INTERDIT POUR LE MOMENT",1);
                      choix_adm = 'x';
                      break;
```

```
/** l'utilisateur a choisi la commande-menu "statistiques" ->  **/
/** demande des parametres de statistiques, dans fen si elle   **/
/** n'est pas occupee (sinon message d'erreur)                 **/
          case 's' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("statistiques");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                "(a)bonne ou (r)apport general : _");
                        nbmot = 3;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "changement de mot  **/
/** de passe" -> demande des parametres de changement de mot de **/
/** passe, dans fen si elle n'est pas occupee (sinon message d' **/
/** erreur                                                     **/
          case 'p' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                    titre("changement du mot de passe d'administration");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                "nouveau mot de passe : _");
                        nbmot = 9;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "changement de cle de **/
/** codage" -> demande des parametres de changement de cle de     **/
/** codage, dans fen si elle n'est pas occupee (sinon message d'  **/
/** erreur                                                       **/
          case 'c' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("changement de la cle de codage");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                "cle de codage          : _");
                        nbmot = 11;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "creation d'un abonne **/
```

```
/** -> demande des premiers parametres de la creation d'un abonne, **/
/** dans fen si elle n'est pas occupee (sinon message d'erreur)    **/
          case 'f' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("creation d'un abonne");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
              "statut ((s)ervice,(p)atron,(se)cretaire,(d)actylo) : _");
                        nbmot = 14;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "suppression d'un abonne" **/
/** -> demande des premiers parametres de la suppression d'un abonne, **/
/** dans fen si elle n'est pas occupee (sinon message d'erreur        **/
          case 'd' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("suppression d'un abonne");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                "identifiant        : _");
                        nbmot = 17;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "modification d'un    **/
/** abonne" -> demande des premiers parametres de la modifica-    **/
/** tion d'un abonne, dans fen si elle n'est pas occupee (sinon   **/
/** message d'erreur)                                             **/
          case 'm' : if (okfen == 1) {
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("modification d'un abonne");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                       "statut ((s)ervice,(se)cretaire,(d)actylo)  : _");
                        nbmot = 19;
                        nbcar = 0;
                        adm = 1;
                        okfen = 0;
                     }
                     else  impmes("  INTERDIT POUR LE MOMENT",1);
                     choix_adm = 'x';
                     break;

/** l'utilisateur a choisi la commande-menu "description d'un     **/
/** abonne -> demande des parametres de la description d'un       **/
/** abonne, dans fen si elle n'est pas occupee (sinon message     **/
/** erreur)                                                       **/
```

```
            case 'i' : if (okfen == 1) {
                           pw_writebackground(fen_pixwin,0,0,900,149,
                                              PIX_CLR);
                           titre("description d'un abonne");
                           win_setmouseposition(fen->ts_windowfd,850,50);
                           pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                   "identifiant : _");
                           nbmot = 23;
                           nbcar = 0;
                           adm = 1;
                           okfen = 0;
                       }
                       else  impmes("  INTERDIT POUR LE MOMENT",1);
                       choix_adm = 'x';
                       break;
        }
}

/*****************************************************************************/

/**** cette fonction est appelee asynchronement chaque fois ****/
/**** qu'un bouton de la souris est enfonce dans fen3 .      ****/

static fen3_input(sw,ibits,ebits,obits,timer)
caddr_t sw;
int *ibits,*ebits,*obits;
struct timeval **timer;

{

/** lecture de l'evenement (un des trois boutons de la souris) **/
      input_readevent(fen3->ts_windowfd,&ievent_fen3);
       *ibits = *obits = *ebits = 0;
       switch (ievent_fen3.ie_code)  {

        case MS_MIDDLE :

/** bouton du milieu **/
                        break;

        case MS_RIGHT  :

/** bouton de droite **/

/** si on est pas dans le bottin (pass-icone = 2), qu'on est  **/
/** quand meme quelque part, que le document pere n'est pas   **/
/** present a l'ecran, et qu'on n'est pas dans consultation   **/
/** administration -> gestion du menu general de consultation **/
/** des resumes                                               **/
                     if ((pass_icone != 2) && (pass_icone != -1)
                         && (pere_pres_doc == 0) && (attente != 1)
                         && (pass_icone != 12) && (pass_icone != 13))
                        ges_menu_cons_res();

/** si on consulte des documents en attente -> gestion du menu **/
/** des documents en attente                                   **/
                     if (attente == 1)
                        ges_menu_attente();

/** si on est dans une consultation administration -> gestion **/
/** du menu administration                                    **/
```

```
                        if (adm_menu == 1)
                           ges_menu_adm();
                        break;

        case MS_LEFT    :

/** bouton de gauche **/
                        break;
        }
}

/*************************************************************************/

/**** cette fonction est appelee asynchronement chaque fois ****/
/**** qu'un bouton de la souris est enfonce dans fen_mes .   ****/

static fen_mes_input(sw,ibits,ebits,obits,timer)
caddr_t sw;
int *ibits,*ebits,*obits;
struct timeval **timer;

{

/** lecture de l'evenement (un des trois boutons de la souris) **/
      input_readevent(fen_mes->ts_windowfd,&ievent_fen_mes);
       *ibits = *obits = *ebits = 0;
       switch (ievent_fen_mes.ie_code)  {

        case MS_MIDDLE :

/** bouton du milieu **/
                        break;

        case MS_RIGHT   :

/** bouton de droite **/
                        break;

        case MS_LEFT    :

/** bouton de gauche **/
                        break;
        }
}

/*************************************************************************/

/*************************************************************/
/******    INITIALISATION DES 2 OPTIONSUBWINDOWS    **********/
/*************************************************************/

/**** initialisation de l'optionsubwindow 1 (celle des icones) ****/

static init_option1()
{
struct item_place place;

        osw = op1->ts_data;

/** initialisation des differentes commandes (icones) de la  **/
/** sous-fenetre op1                                         **/
```

```
        c1o1_item = optsw_command(osw,&icon1_label,c1o1_proc);
        c2o1_item = optsw_command(osw,&icon2_label,c2o1_proc);
        c3o1_item = optsw_command(osw,&icon3_label,c3o1_proc);
        c4o1_item = optsw_command(osw,&icon4_label,c4o1_proc);
        c5o1_item = optsw_command(osw,&icon5_label,c5o1_proc);
        c6o1_item = optsw_command(osw,&icon6_label,c6o1_proc);
        c7o1_item = optsw_command(osw,&icon7_label,c7o1_proc);
        c8o1_item = optsw_command(osw,&icon8_label,c8o1_proc);
        c9o1_item = optsw_command(osw,&icon9_label,c9o1_proc);
        c10o1_item = optsw_command(osw,&icon10_label,c10o1_proc);
        c11o1_item = optsw_command(osw,&icon11_label,c11o1_proc);
        c12o1_item = optsw_command(osw,&icon12_label,c12o1_proc);
        c13o1_item = optsw_command(osw,&icon13_label,c13o1_proc);

/** placement des icones dans la sous-fenetre des icones op1 **/
        place.rect.r_top = 7;
        place.rect.r_height = 64;
        place.rect.r_width = 64;
        place.fixed.x = place.fixed.y = 1;
        place.fixed.w = place.fixed.h = 1;

        place.rect.r_left = 22;
        optsw_setplace(osw,c1o1_item,&place,FALSE);
        place.rect.r_left = 108;
        optsw_setplace(osw,c2o1_item,&place,FALSE);
        place.rect.r_left = 194;
        optsw_setplace(osw,c3o1_item,&place,FALSE);
        place.rect.r_left = 280;
        optsw_setplace(osw,c4o1_item,&place,FALSE);
        place.rect.r_left = 366;
        optsw_setplace(osw,c5o1_item,&place,FALSE);
        place.rect.r_left = 452;
        optsw_setplace(osw,c6o1_item,&place,FALSE);
        place.rect.r_left = 538;
        optsw_setplace(osw,c7o1_item,&place,FALSE);
        place.rect.r_left = 624;
        optsw_setplace(osw,c8o1_item,&place,FALSE);
        place.rect.r_left = 710;
        optsw_setplace(osw,c9o1_item,&place,FALSE);
        place.rect.r_left = 796;
        optsw_setplace(osw,c10o1_item,&place,FALSE);
        place.rect.r_left = 882;
        optsw_setplace(osw,c11o1_item,&place,FALSE);
        place.rect.r_left = 968;
        optsw_setplace(osw,c12o1_item,&place,FALSE);
        place.rect.r_left = 1054;
        optsw_setplace(osw,c13o1_item,&place,FALSE);
}

/*****************************************************************/

/**** initialisation de l'optionsubwindow 2 (celle des commandes) ****/

static init_option2()
{

/** structure designant un endroit dans un optionsubwindow **/
struct item_place place;

        osw = op2->ts_data;
```

```
/** initialisation des commandes de la sous-fenetre op2 **/
        c1o2_item = optsw_command(osw,&c1o2,c1o2_proc);
        c2o2_item = optsw_command(osw,&c2o2,c2o2_proc);
        c3o2_item = optsw_command(osw,&c3o2,c3o2_proc);
        c4o2_item = optsw_command(osw,&c4o2,c4o2_proc);

/** mise en place de la commande consultation **/
        place.rect.r_left = optsw_coltox(osw,2);
        place.rect.r_top = 49;
        place.rect.r_height = optsw_linetoy(osw,1);
        place.rect.r_width = optsw_coltox(osw,11);
        place.fixed.x = place.fixed.y = 1;
        place.fixed.w = place.fixed.h = 1;
        optsw_setplace(osw,c1o2_item,&place,FALSE);

/** mise en place de la commande editer **/
        place.rect.r_left = optsw_coltox(osw,16);
        place.rect.r_top = 49;
        place.rect.r_height = optsw_linetoy(osw,1);
        place.rect.r_width = optsw_coltox(osw,8);
        place.fixed.x = place.fixed.y = 1;
        place.fixed.w = place.fixed.h = 1;
        optsw_setplace(osw,c2o2_item,&place,FALSE);

/** mise en place de la commande transfert **/
        place.rect.r_left = optsw_coltox(osw,2);
        place.rect.r_top = 105;
        place.rect.r_height = optsw_linetoy(osw,1);
        place.rect.r_width = optsw_coltox(osw,12);
        place.fixed.x = place.fixed.y = 1;
        place.fixed.w = place.fixed.h = 1;
        optsw_setplace(osw,c3o2_item,&place,FALSE);

/** mise en place de la commande quitter **/
        place.rect.r_left = optsw_coltox(osw,16);
        place.rect.r_top = 105;
        place.rect.r_height = optsw_linetoy(osw,1);
        place.rect.r_width = optsw_coltox(osw,9);
        place.fixed.x = place.fixed.y = 1;
        place.fixed.w = place.fixed.h = 1;
        optsw_setplace(osw,c4o2_item,&place,FALSE);

}

/*****************************************************************************/
/*****************   PROCEDURES  ICONES   ********************************/
/*****************************************************************************/

/**** initialise le temoin de passage dans les icones ****/

static init_pass_icone()
{
        pass_icone = 0;
}

/*****************************************************************************/

/**** gestion de l'icone boite out ****/

static c1o1_proc()
{
```

```
char *text1 = "nom du destinataire : \0";
char *text2 = "titre du document   : \0";
char *text3 = "redacteur           : \0";
char *text5 = "confidentiel (o/n)  : \0";
char *text6 = "rappel (o/n)        : \0";
char *text7 = "recommande (o/n)    : \0";
char *text8 = "reponse (o/n)       : \0";
char *text4 = "mots-cles           : \0";

        switch (pass_commande) {

/** si le temoin de commande est a consultation (commande 1) -> **/
/** message d'erreur                                           **/
            case 1 : impmes("CONSULTATION BOITE OUT",2);
                     impmes("  PAS DE CONSULTATION DE BOITE OUT",1);
                     break;

/** si le temoin de commande est a transfert (commande 2) ->  **/
/** preparation de fen pour la saisie des parametres d'envoi, **/
/** initialisation du nombre de caracteres et de mots tapes,  **/
/** du nombre de parametres a demander et mise a 1 du temoin  **/
/** de presence dans la saisie des parametres d'envoi         **/
            case 2 : impmes("TRANSFERT BOITE OUT",2);
                     if (get_perm(1) == 1) {
                      if (okfen == 1) {
                        okfen = 0;
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("parametres d'envoi");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,text1);
                        pw_char(fen_pixwin,
                                30 + 21*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,'_');
                        pw_text(fen_pixwin,20,65,PIX_SRC,NULL,text2);
                        pw_text(fen_pixwin,20,85,PIX_SRC,NULL,text3);
                        pw_text(fen_pixwin,20,105,PIX_SRC,NULL,text5);
                        pw_text(fen_pixwin,20,125,PIX_SRC,NULL,text6);
                        pw_text(fen_pixwin,20,145,PIX_SRC,NULL,text7);
                        pw_text(fen_pixwin,470,45,PIX_SRC,NULL,text8);
                        pw_text(fen_pixwin,470,65,PIX_SRC,NULL,text4);
                        nb_mot_cles = 0;
                        saisie_param = 1;
                        au_moins_un = 0;
                        nbcar = 0;
                        nbmot = 1;
                        rap = 8;
                      }
                      else impmes("  INTERDIT POUR LE MOMENT",1);
                     }
                     else impmes("  TRANSFERT INTERDIT",1);
                     break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  BOITE OUT SEULE",1);
        }
        init_pass_commande();
}

/*****************************************************************/
```

```
/**** gestion de l'icone bottin ****/

static c2o1_proc()
{
int i;

       switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 1) -> **/
/** message d'erreur                                        **/
              case 2 : impmes("TRANSFERT BOTTIN",2);
                       impmes("  PAS DE TRANSFERT VERS BOTTIN",1);
                       break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** initialisation de cont_fen2, impression de la liste des     **/
/** listes dans fen2 et remplissage de cont_fen2 et mise a      **/
/** bottin du temoin de passage dans les icones (icone 2)       **/
              case 1 : impmes("CONSULTATION BOTTIN",2);
                       preslist = 1;
                       pere_pres_res = 0;
                       pere_pres_doc = 0;
                       pw_writebackground(fen2_pixwin,0,0,567,600,
                                          PIX_CLR);
                       pw_writebackground(fen3_pixwin,0,0,570,600,
                                          PIX_CLR);
                       init_tab_cont_fen2();
                       win_setmouseposition(fen2->ts_windowfd,200,200);
                       for (i=0;i<lg_tabbot+newlist;++i)  {
                       pw_text(fen2_pixwin,20,30*(i+1),PIX_SRC,
                               NULL,tabbot[i]);
                       put_tab_cont_fen2(30*(i+1) -
                                         font->pf_defaultsize.y,
                                         tabbot[i]);
                       }
                       pass_icone = 2;
                       break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
              default : impmes("  BOTTIN SEUL",1);
       }
       init_pass_commande();
}

/*****************************************************************/

/**** gestion de l'icone poubelle ****/

static c3o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 1) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
              case 2 : impmes("TRANSFERT POUBELLE",2);
                       if (get_perm(3)==1) {
```

```
                        impmes("  TRANSFERT EFFECTUE",1);
                    }
                    else impmes("  TRANSFERT INTERDIT",1);
                    break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** impression du premier document de la collection des docu-   **/
/** ments en poubelle et impression de son resume               **/
            case 1 : impmes("CONSULTATION POUBELLE",2);
                    preslist = 0;
                    pass_icone = 3;
                    pere_pres_res = 0;
                    pere_pres_doc = 0;
                    pw_writebackground(fen2_pixwin,0,0,565,600,
                                       PIX_CLR);
                    pw_writebackground(fen3_pixwin,0,0,570,600,
                                       PIX_CLR);
                    fpres = fopen("res","r");
                    imprime(fpres,1,3);
                    fclose(fpres);
                    fpdoc = fopen("doc","r");
                    imprime(fpdoc,1,2);
                    fclose(fpdoc);
                    break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
            default : impmes("  POUBELLE SEULE",1);
        }
        init_pass_commande();
}
/*****************************************************************/

/**** gestion de l'icone archives ****/

static c4o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 1) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
            case 2 : impmes("TRANSFERT ARCHIVES",2);
                    if (get_perm(4)==1) {
                       impmes("  TRANSFERT ARCHIVES EFFECTUE",1);
                    }
                    else impmes("  TRANSFERT INTERDIT",1);
                    break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** demande des parametres de consultation des archives, dans   **/
/** fen si elle n'est pas occupee (sinon message d'erreur)      **/
            case 1 : impmes("CONSULTATION ARCHIVES",2);
                    if (okfen == 1) {
                       okfen = 0;
                       preslist = 0;
                       pass_icone = 4;
                       pere_pres_res = 0;
                       pere_pres_doc = 0;
```

```
                        pw_writebackground(fen2_pixwin,0,0,565,600,
                                           PIX_CLR);
                        pw_writebackground(fen3_pixwin,0,0,570,600,
                                           PIX_CLR);
                        pw_writebackground(fen_pixwin,0,0,900,149,
                                           PIX_CLR);
                        titre("selection de documents archives");
                        win_setmouseposition(fen->ts_windowfd,850,50);
                        pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                "document (e)nvoye ou (r)ecu : ");
                        pw_char(fen_pixwin,
                                20+30*font->pf_defaultsize.x,
                                45,PIX_SRC,NULL,'_');
                        archive = 1;
                        nbcar = 0;
                        nbmot = 1;
                     }
                     else impmes("  INTERDIT POUR LE MOMENT",1);
                     break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                            **/
                default : impmes("  ARCHIVES SEULES",1);
          }
          init_pass_commande();
}

/*****************************************************************************/

/**** gestion de l'icone documents en attente ****/

static c5o1_proc()
{

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 1) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("TRANSFERT ATTENTE",2);
                         if (get_perm(5)==1) {
                            impmes("  TRANSFERT ATTENTE EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** message disant que le menu concernant les documents en      **/
/** attente peut etre active                                    **/
                case 1 : impmes("CONSULTATION ATTENTE",2);
                         pere_pres_res = 0;
                         pere_pres_doc = 0;
                         pw_writebackground(fen3_pixwin,0,0,570,600,
                                            PIX_CLR);
                         pw_writebackground(fen2_pixwin,0,0,565,600,
                                            PIX_CLR);
                         pw_text(fen3_pixwin,20,20,PIX_SRC,NULL,
                    "veuillez activer le menu avec le bouton de droite");
                         attente = 1;
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
```

```
/** -> message d'erreur                                                **/
            default : impmes("  ATTENTE SEULE",1);
        }
        init_pass_commande();
}

/*********************************************************************/

/**** gestion de l'icone printer ****/

static c6o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a consultation (commande 1) -> **/
/** message d'erreur                                           **/
            case 1 : impmes("  PAS DE CONSULTATION IMPRIMANTE",1);
                     impmes("CONSULTATION IMPRIMANTE",2);
                     break;

/** si le temoin de commande est a transfert (commande 2) ->   **/
/** impression du document selectionne avec message de con-    **/
/** firmation, ou message d'interdiction                       **/
            case 2 : impmes("TRANSFERT IMPRIMANTE",2);
                     if (get_perm(6) == 1) {
                      if (doc_select != 0) {
                         fpdoc_print = fopen("doc_print","w");
                         fpdoc = fopen("doc","r");
                         imprime(fpdoc,doc_select,0);
                         fclose(fpdoc);
                         fclose(fpdoc_print);
                         system("cat doc_print>/dev/ttyp0");
                      }
                      else system("cat nouv>/dev/ttyp0");
                     }
                     else impmes("  TRANSFERT INTERDIT",1);
                     break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
            default : impmes("  IMPRIMANTE SEULE",1);
        }
        init_pass_commande();
}

/*********************************************************************/

/**** gestion de l'icone boite in ****/

static c7o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
```

```
                case 2 : impmes("TRANSFERT BOITE IN",2);
                         if (get_perm(7)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** impression du premier document de la collection des docu-   **/
/** ments de la boite aux lettres et impression de son resume   **/
                case 1 : impmes("CONSULTATION BOITE IN",2);
                         preslist = 0;
                         pass_icone = 7;
                         pere_pres_res = 0;
                         pere_pres_doc = 0;
                         pw_writebackground(fen2_pixwin,0,0,565,600,
                                            PIX_CLR);
                         pw_writebackground(fen3_pixwin,0,0,570,600,
                                            PIX_CLR);
                         fpres = fopen("res","r");
                         imprime(fpres,1,3);
                         fclose(fpres);
                         fpdoc = fopen("doc","r");
                         imprime(fpdoc,1,2);
                         fclose(fpdoc);
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  BOITE IN SEULE",1);
        }
        init_pass_commande();
}

/*****************************************************************/

/**** gestion de l'icone signataire ****/

static c8o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("TRANSFERT SIGNATAIRE",2);
                         if (get_perm(8)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** impression du premier document de la collection des docu-   **/
/** ments du signataire et impression de son resume             **/
                case 1 : impmes("CONSULTATION SIGNATAIRE",2);
                         preslist = 0;
                         pass_icone = 8;
                         pere_pres_res = 0;
```

```
                        pere_pres_doc = 0;
                        pw_writebackground(fen2_pixwin,0,0,565,600,
                                           PIX_CLR);
                        pw_writebackground(fen3_pixwin,0,0,570,600,
                                           PIX_CLR);
                        fpres = fopen("res","r");
                        imprime(fpres,1,3);
                        fclose(fpres);
                        fpdoc = fopen("doc","r");
                        imprime(fpdoc,1,2);
                        fclose(fpdoc);
                        break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  SIGNATAIRE SEUL",1);
        }
        init_pass_commande();
}

/*****************************************************************************/

/**** gestion de l'icone poste ****/

static c9o1_proc()
{
struct rect r1;
int i;

        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("TRANSFERT POSTE",2);
                         if (get_perm(9)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** impression de la date d'envoi du document selectionne       **/
                case 1 : impmes ("CONSULTATION POSTE",2);
                         if (doc_select != 0) impmes("  12 mars 1985");
                         else impmes("  PAS DE DOCUMENT SELECTIONNE",1);
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  POSTE SEULE",1);
        }
        init_pass_commande();
}

/*****************************************************************************/

/**** gestion de l'icone mot de passe ****/

static c10o1_proc()
{
```

```
        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("TRANSFERT MOT DE PASSE",2);
                         if (get_perm(10)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** demande des premiers parametres du changement de mot de     **/
/** passe et initialisation des variables necessaires a ce      **/
/** changement                                                  **/
                case 1 : impmes("CONSULTATION MOT DE PASSE",2);
                         if (okfen == 1) {
                            okfen = 0;
                            pere_pres_res = 0;
                            pere_pres_doc = 0;
                            mp = 1;
                            pw_writebackground(fen_pixwin,0,0,900,149,
                                               PIX_CLR);
                            win_setmouseposition(fen->ts_windowfd,850,50);
                            titre("changement de mot de passe");
                            pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                    "mot de passe d'abonne : ");
                            pw_char(fen_pixwin,
                                    30 + 24*font->pf_defaultsize.x,
                                    45,PIX_SRC,NULL,'_');
                            nbcar = 0;
                            newnbmot = 1;
                            nbmot = 1;
                            nb_mot_passe_false = 0;
                            strcpy(new_mot_passe1,"\0");
                            strcpy(new_mot_passe2,"\0");
                         }
                         else impmes("  INTERDIT POUR LE MOMENT",1);
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  MOT DE PASSE SEUL",1);
        }
        init_pass_commande();
}

/*****************************************************************/

/**** gestion de l'icone reponses recues ****/

static c11o1_proc()
{
struct rect r1;
int i;

     if (doc_select != 0) {
        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
```

```
                case 2 : impmes("TRANSFERT REPONSES RECUES",2);
                         if (get_perm(11)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) ->    **/
/** impression du premier document de la collection des reponses  **/
/** recues et impression de son resume                            **/
                case 1 : impmes("CONSULTATION REPONSES RECUES",2);
                         preslist = 0;
                         pass_icone = 11;
                         pere_pres_res = 0;
                         pere_pres_doc = 0;
                         pw_writebackground(fen2_pixwin,0,0,565,600,
                                            PIX_CLR);
                         pw_writebackground(fen3_pixwin,0,0,570,600,
                                            PIX_CLR);
                         fpres = fopen("res","r");
                         imprime(fpres,1,3);
                         fclose(fpres);
                         fpdoc = fopen("doc","r");
                         imprime(fpdoc,1,2);
                         fclose(fpdoc);
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                            **/
                default : impmes("  REPONSES RECUES SEULES",1);
            }
            init_pass_commande();
        }
        else impmes("  PAS DE DOCUMENT SELECTIONNE",1);
}

/*****************************************************************/

/**** gestion de l'icone pere ****/

static c12o1_proc()
{

     if (doc_select != 0) {
        switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("TRANSFERT PERE",2);
                         if (get_perm(12)==1) {
                            impmes("  TRANSFERT EFFECTUE",1);
                         }
                         else impmes("  TRANSFERT INTERDIT",1);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** impression du document pere du document selectionne         **/
                case 1 : impmes("CONSULTATION PERE",2);
                         preslist = 0;
                         pere_pres_res = 0;
                         pere_pres_doc = 0;
```

```
                        pass_icone = 12;
                        pw_writebackground(fen2_pixwin,0,0,565,600,
                                           PIX_CLR);
                        pw_writebackground(fen3_pixwin,0,0,570,600,
                                           PIX_CLR);
                        fppere = fopen("pere","r");
                        imprime(fppere,1,2);
                        fclose(fppere);
                        break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  PERE SEUL",1);
            }
            init_pass_commande();
        }
        else impmes("  PAS DE DOCUMENT SELECTIONNE",1);
}

/*****************************************************************************/

/**** gestion de l'icone administration ****/

static c13o1_proc()
{

            attente = 0;
            switch (pass_commande) {

/** si le temoin de commande est a transfert (commande 2) -> **/
/** message de confirmation ou d'interdiction de transfert   **/
                case 2 : impmes("  TRANSFERT INTERDIT",1);
                         impmes("TRANSFERT ADMINISTRATION",2);
                         break;

/** si le temoin de commande est a consultation (commande 1) -> **/
/** demande du mot de passe d'administration                    **/
                case 1 : impmes("CONSULTATION ADMINISTRATION",2);
                         if (okfen == 1) {
                            okfen = 0;
                            pw_writebackground(fen_pixwin,0,0,900,149,
                                               PIX_CLR);
                            win_setmouseposition(fen->ts_windowfd,850,50);
                            pw_text(fen_pixwin,20,45,PIX_SRC,NULL,
                                    "mot de passe d'administration : _");
                            nbcar = 0;
                            nbmot = 1;
                            adm = 1;
                            preslist = 0;
                            pere_pres_res = 0;
                            pere_pres_doc = 0;
                            pass_icone = 13;
                         }
                         else impmes("  INTERDIT POUR LE MOMENT",1);
                         break;

/** si le temoin de commande n'est ni a transfert ni a consultation **/
/** -> message d'erreur                                             **/
                default : impmes("  ADMINISTRATION SEUL",1);
            }
            init_pass_commande();
```

```
}

/*****************************************************************************/
/*******************   PROCEDURES   COMMANDES   ******************************/
/*****************************************************************************/

/**** initialisation du temoin de passage des commandes ****/

static init_pass_commande()
{
        pass_commande = 0;
}

/*****************************************************************************/

/**** gestion de la commande consultation (commande 1) ****/

static c1o2_proc()
{

/** si on est en train d'editer ou de donner les parametres **/
/** d'un rappel -> erreur                                   **/
        if ((edit_rappel == 0) && (att_rep == 0) && (adm == 0)
            && ((okfen == 1) || ((okfen == 0) && (pass_icone == 2)))) {
           numres = 1;
           numdoc = 1;
           pass_commande = 1;
           win_setmouseposition(op1->ts_windowfd,10,10);
           attente = 0;
           adm_menu = 0;
        }
        else impmes("  CONSULTATION INTERDITE POUR LE MOMENT",1);
}

/*****************************************************************************/

/**** gestion de la commande editer (commande 2) ****/

static c2o2_proc()
{
int i;
struct rect rr1,rr2;

        if ((edit_en_cours == 0) &&
            (edit_rappel == 0)   &&
            (edit_corps == 0)    &&
            (edit_rep == 0)      &&
            (edit_mod == 0)      &&
            (edit_note == 0)) {

/** on n'est pas dans une edition -> on peut editer, creation d'un **/
/** processus qui execute un 'vi nouv'                             **/
           rect_construct(&rr1,0,0,0,0);
           rect_construct(&rr2,0,0,0,0);
           wmgr_forktool("e4\0","nouv 3\0",&rr1,&rr2,0);
           edit_en_cours = 1;
           doc_select = 0;
           put_perm(0);
        }
        else {
```

```
/** on est dans une edition -> on ne peut pas editer, message **/
/** d'erreur                                                  **/
        impmes("  EDITION INTERDITE DANS UNE EDITION",1);
        }
}

/*****************************************************************/

/**** gestion de la commande transfert ****/

static c3o2_proc()
{

/** si fen est libre (okfen = 1) -> pass_commande = 2 (c-a-d un  **/
/** transfert)                                                   **/
        if (okfen == 1) {
           attente = 0;
           adm_menu = 0;
           pass_commande = 2;
           win_setmouseposition(op1->ts_windowfd,10,10);
        }

/** si fen n'est pas libre (okfen = 0) -> message d'erreur   **/
        else impmes("  TRANSFERT INTERDIT POUR LE MOMENT",1);
}

/*****************************************************************/

/**** gestion de la commande quitter ****/

static c4o2_proc()
{
char *text;

/** impression d'un message de confirmation et destruction du   **/
/** tool s'il y a confirmation                                  **/
        text = "appuyer sur le bouton de gauche pour confirmer";
        if (wmgr_confirm(tooll->tl_windowfd,text) == -1) {
            tool_destroy(tooll);
            exit(2);
        }
}

/*****************************************************************/

/****  imprime dans la fenetre designee par fenetre, le document ****/
/****  de numero place se trouvant dans le fichier nom           ****/

static imprime(nom,place,fenetre)
FILE *nom;
int place,fenetre;
{
int i,j,nl;
char c;

        i = 1;
        nl = 1;
        j = 0;

/** recherche le bon document dans le bon fichier  **/
        while (i<place) if (getc(nom)=='@') ++i;
```

```
/** imprime au bon endroit le document trouve, soit dans une sous- **/
/** fenetre, soit dans un autre fichier                            **/
       while ((c=getc(nom))!='@')
             if (c=='\n') {

/** impression dans les fichiers doc_print, doc_mod, doc_corps     **/
/** suivant la valeur de la variable fenetre                       **/
                 if (fenetre==0) putc(c,fpdoc_print);
                 if (fenetre==1) putc(c,fpdoc_mod);
                 if (fenetre==4) putc(c,fpdoc_corps);
                 ++nl;
                 j = 0;
             }
             else {

/** impression dans fen2  **/
                  if (fenetre==2) pw_char(fen2_pixwin,
                                          20+(j*font->pf_defaultsize.x),
                                          20*nl,PIX_SRC,NULL,c);

/** impression dans fen3  **/
                  else if (fenetre == 3)
                          pw_char(fen3_pixwin,
                                  20+(j*font->pf_defaultsize.x),
                                  20*nl,PIX_SRC,NULL,c);
                       else if (fenetre == 0) putc(c,fpdoc_print);
                            else if (fenetre == 1) putc(c,fpdoc_mod);
                                 else putc(c,fpdoc_corps);
                  ++j;
             }
}

/*****************************************************************/

/**** ****/

static put_perm(num)
int num;
{
int i;

/** initialisation du tableau de permission des transferts  **/
       for (i=0;i<10;++i) tabperm[i] = 0;

       switch (num) {

/** remplissage du tableau de permission des transferts en fonction  **/
/** des icones : 1 = transfert permis; 0 = transfert interdit        **/
              case 0 : for(i=1;i<11;++i) tabperm[i] = 1;
                       break;

              case 2 : break;

              case 3 : tabperm[5] = 1;
                       tabperm[4] = 1;
                       tabperm[1] = 1;
                       tabperm[6] = 1;
                       break;

              case 4 : tabperm[6] = 1;
```

```
                        tabperm[1] = 1;
                        break;

                case 51: tabperm[4] = 1;
                        tabperm[6] = 1;
                        break;

                case 52: tabperm[6] = 1;
                        tabperm[4] = 1;
                        break;

                case 53: tabperm[6] = 1;
                        tabperm[4] = 1;
                        tabperm[1] = 1;
                        tabperm[3] = 1;
                        break;

                case 6 : break;

                case 7 : tabperm[5] = 1;
                        tabperm[4] = 1;
                        tabperm[1] = 1;
                        tabperm[6] = 1;
                        tabperm[3] = 1;
                        break;

                case 8 : tabperm[6] = 1;
                        break;

                case 9 : break;

                case 10 : break;

                case 11 : tabperm[6] = 1;
                        break;

                case 12 : tabperm[6] = 1;
                        tabperm[1] = 1;
                        break;
        }
}

/*****************************************************************/

/****  renvoi la partie du tableau de permisison des transfert  ****/
/****  concernant l'icone de numero num                         ****/

static get_perm(num)
int num;
{

        return(tabperm[num]);
}

/*****************************************************************/
/*****************************************************************/
/************************ FIN DU PROGRAMME ***********************/
/*****************************************************************/
/*****************************************************************/
```